UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Technical University
of Denmark

Máster Universitario en Bioinformática
y Biología Computacional

TRABAJO FIN DE MÁSTER

Understanding how cancer mutations hinder the interactions inside proteins

Autor: Carmen Sáenz Ausejo

Tutor: Jose María Gónzalez Izarzugaza
Ponente: Luis Del Peso Ovalle

FEBRUARY, 2018

Understanding how cancer mutations hinder the interactions inside proteins

Identification of 3D protein clusters as potential functional targets applied to BRCA-mutated breast cancer patients

Autor: Carmen Sáenz Ausejo

Tutor: Jose María Gónzalez Izarzugaza
Bio and Health Informatics Department, DTU
Ponente: Luis Del Peso Ovalle
Biochemistry Department, UAM

FEBRUARY, 2018

Summary

The acquisition of somatic mutations can induce cancer by **dysregulating the delicate mechanisms controlling balance between** proliferation and apoptosis. **Genomic alterations can be classified in** driver and passenger mutations.

Driver mutations confer selective advantage to tumor development, contrarily to *passenger* mutations that do not provide growth advantage to tumorigenesis. Most of the driver mutations have unknown functional impact on protein structure and function. Furthermore, **not all driver alterations in a cancer gene have the same functional impact**.

The use of high-throughput sequencing technologies facilitated the discovery of cancer related mutations in case and control studies. The analysis of different tumor types facilitates the identification of recurrent mutations and the functional pathways involved in tumor development.

One of the current challenges is to distinguish between drivers and passenger mutations. Mutations occurring with high frequency in tumor samples are considered to be drivers. Therefore, a commonly used method is to consider mutations that occur with higher frequency than a background mutation rate.

Tamborero *et al.*, (2013) developed a method to identify cancer related genes by grouping together residues with a significant rate of mutations that are close in the primary sequence of the protein above the background model. The background model was generated considering coding-silent mutations based on the evidences of a nonrandom mutation processes along the genome (Amos, 2010).

Recently, Gao et al., (2017) identified genomic mutations affecting residues located in 3-dimensional proximity of protein structures by comparing the mutation frequency against a random background.

The first method used gene sequences, considering proteins as single strands, and omitted that distant genomic regions might be close in the 3D space when the protein folds. And the second method assumed a homogeneous mutation probability across the whole genome, which is likely an oversimplification that may introduces a bias in the expected mutation rates (Amos, 2010).

Both problems were considered in this study for the development of the algorithm. This method identifies associated with BRCA-mutated breast cancer using coding-silent

mutation frequency as a background. Furthermore, the method identified structural and catalytic roles of 3D protein clusters within relevant biological pathways in breast cancer.

This method considered that a 3D protein cluster is significant when the residues within it have a higher non-synonymous mutation rate as compared to the background mutation rate.

Most of the significant 3D protein clusters were located within *PIK3CA* gene. Additionally, most of the mutations in the 3D clusters were predominantly found in the kinase and helical domains of the corresponding protein (PI3K). These mutations destabilize the inactive conformation of the proteins or lock the activation loop in an active conformation resulting in constitutive protein activation. Thus, significant 3D protein clusters in *PIK3CA* contain ideal hot-spot mutants to target with anti-cancer agents (Gabelli, Mandelker, Schmidt-Kittler, Vogelstein, & Amzel, 2010).

Nowadays, treatments with PI3K inhibitors are available. However, the oncogenic PI3K pathway activation is achieved in different redundant ways, therefore mono-therapies are not always effective.

In conclusion, the results of this Master´s Thesis can help to understand better the interactions of the non-synonymous mutations in the 3D protein space to identify new targets, develop new therapies and consequently maximize the therapeutic benefit.

Key words: Breast cancer, driver mutations, non-synonymous mutations, synonymous background model, 3D protein clusters

Acknowledgments

I would like to express my deepest appreciation to all those who provided me the possibility to complete this research. And I would like to thank you all the people that supported me during my Master degree, my family, my friends and my teachers.

Dr. Jose M.G. Izarzugaza under whom I´d been working the past 4 months and DTU Bio and Health Informatics Department where I did my Masters Thesis, for letting me participate and helped me to carry out this project.

Gianluca Mazzoni has been an important support for me during the entire project, giving me references and corrections to improve my work.

Also, to the people in the office with whom I worked everyday, particularly, Diego Calvo that had always time to help me and discuss with me doubts.

Thank you all for giving me the support that I needed and doing the best to help me.

Index

Objectives

The main goal of this study is to identify 3D protein clusters, rather than cancer-related genes, containing non-synonymous mutations and annotate them.

The first objective is to understand how cancer mutations hinder the interaction inside proteins using synonymous mutations as a background model to identify non-synonymous mutations grouped in 3D spatial clusters as potential functional targets in BRCA-mutated breast cancer.

Once the significant ones are selected, the second objective is to annotate driver clusters with a structural or catalytic role within the protein.

Introduction

Mutations

Permanent alterations of the nucleotide sequence of the genome are called **mutations**. These can occur during the replication or repair processes, in either somatic or germ-line cells. **Somatic mutations** are not inherited, so they can be disregarded in an evolutionary or genetic context (Learn Science at Scitable, 2018).

Mutations may be classified (Fig.1) by the length of the altered DNA sequence. Here, we will focus on those affecting a single nucleotide (*single-base* or point mutations), specifically, on *substitution mutations*, the replacement of one nucleotide by another.

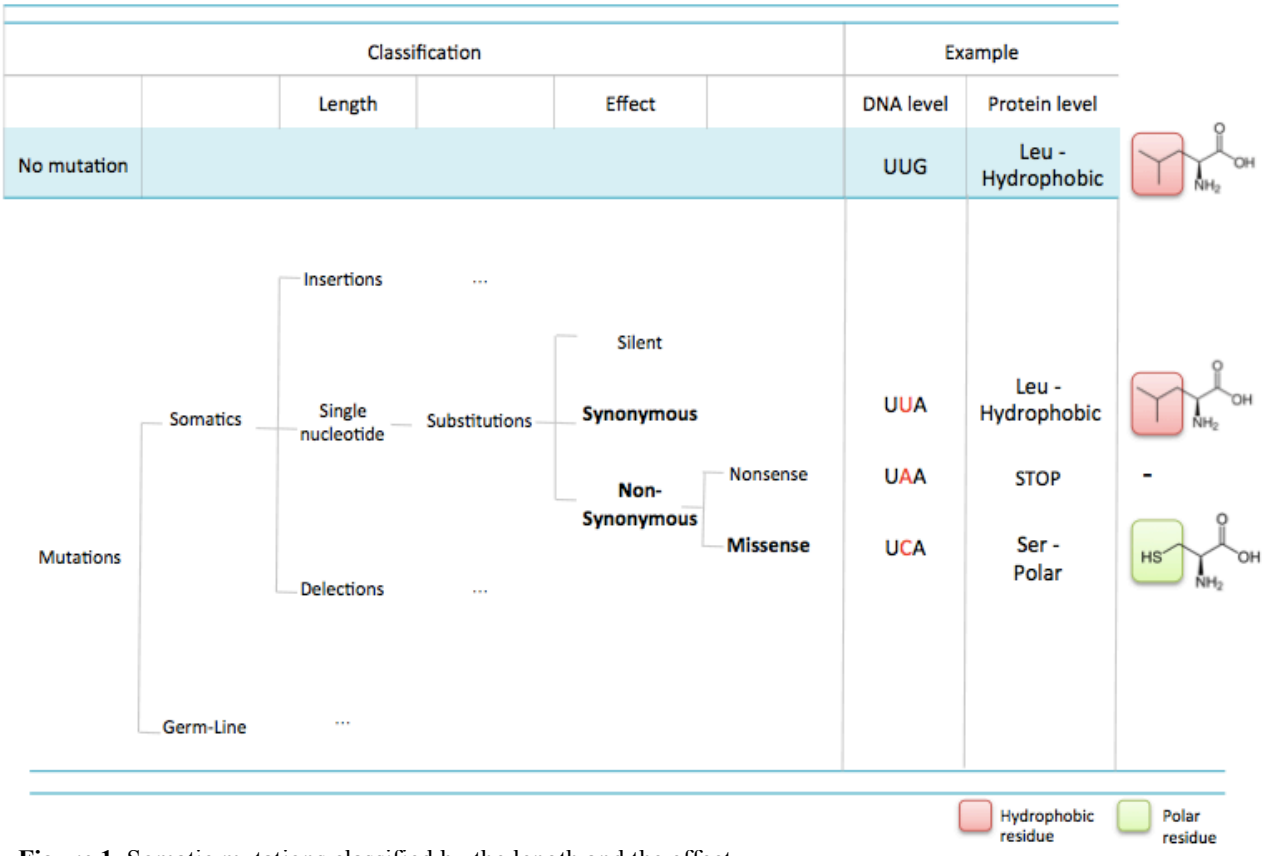


Figure 1. Somatic mutations classified by the length and the effect.

Those substitution mutations occurring in protein-coding regions may also be classified (Fig.1) according to their effect on the protein. Because the genetic code is degenerate, most amino acids are represented by more than one triplet of nucleotide bases. These alternative codes for the same amino acid are called synonymous or silent codons.

Silent mutations are mutations that do not result in a change to the amino acid

sequence of a protein but do change the nucleotide sequence, unless the changed amino-acid is sufficiently similar to the original. They may occur in a region that does not code for a protein, or within a codon in a manner that does not alter the final amino acid sequence. While s**ynonymous mutations** (Fig.1) occur only within exons and are not always silent mutations.

The terms "synonymous" and "silent" mutation are often used interchangeably because, in the great majority of cases, synonymous mutations do not alter the amino acid sequence of a protein and are therefore not detectable at the amino acid level.

Nonsynonymous mutations (amino acid-altering mutations) (Fig.1) are single nucleotide changes that cause substitutions of different amino acids, resulting in abnormal protein products (Learn Science at Scitable, 2018). Are classified into *missense* and *nonsense mutations*. The former one changes the affected codon into a codon that specifies a different amino acid from the one previously encoded. While the latter one, changes a sense codon into a termination codon. These mutations can cause the resulting protein nonfunctional.

Early somatic mutations can cause developmental disorders, whereas the progressive accumulation of mutations (P. J. Campbell, Martinocorena, & Campbell, 2015)**,** many of which regulate cell division, enable cells enter a state of uncontrolled division, resulting in a cluster of cells called **tumor.**

Tumorigenesis can be due to alterations in three types of genes: oncogenes, tumor-suppressor and stability genes, that control cell proliferation, differentiation and cellular homeostasis (Vogelstein & Kinzler, 2004).

- Oncogenes, oncogenic variants of the normal proto-oncogenes that acquired a gain-of-function alteration, encode proteins that control cell proliferation, apoptosis, or both (Croce, 1995). A somatic mutation is able to cause either an alteration in the oncogene structure or an increase in or deregulation of its expression (Bishop, 1991), and therefore to confer a selective growth advantage on the cell (i.e.: PIK3CA)

- Tumor-suppressor genes play important roles in suppressing uncontrolled proliferation, immortality, and tumorigenicity (i.e.:*TP53*, *PTEN*) (Guo, Ngo, Modrek, & Lee, 2014). Mutations reduce the activity of the gene product.

- Stability genes or caretakers, promotes tumorigenesis in a completely different way when mutated. Include genes responsible for repairing subtle mistakes made during normal DNA replication or control processes involving large portions of chromosomes, such as those responsible for mitotic recombination and chromosomal segregation (i.e.: *BLM* and *ATM*) (Vogelstein & Kinzler, 2004). Stability genes keep genetic alterations to a minimum, and thus when they are inactivated, mutations in other genes occur at a higher rate.

The first somatic mutation in an oncogene or tumor-suppressor gene that causes a clonal expansion initiates the neoplastic process (Nowell, 2002). Subsequent somatic mutations result in additional rounds of clonal expansion and thus in tumor progression (Maley et al., 2004; Vogelstein & Kinzler, 2004).

All genes are potentially affected by the resultant increased rate of mutation. But only can confer a selective growth advantage to the mutant cell those mutations which effect is the overexpression of oncogenes and the loss of tumor suppressors (Nowell, 2002). Mutations in these genes are the dominant driving forces for tumorigenesis. Hence, targeting oncogenes and tumor suppressors hold tremendous therapeutic potential for cancer treatment.

A subset of these somatic alterations, termed **driver mutations**, confer selective growth advantage within a population of cells and are implicated in cancer development. Whereas the remainder, **passengers mutations**, have either no phenotypic consequences or biological effects that are not selectively advantageous to the clone (Stratton, Campbell, & Futreal, 2009), thus do not contribute to oncogenesis (Pleasance et al., 2010).

The existence of different types of genes and mutations in cancer has significant implications for developing targeted therapies in cancer care (Ye, Pavlicek, Lunney, Rejto, & Teng, 2010).

Hot-spots

While each human tumor has its own unique "genetic signature," certain oncogenes are often found mutated at the same unique positions. These so called **"hot-spot" mutations** can characterize a specific cancer subtype, confer resistance or sensitivity to individual inhibitors, and in some cases, correlate with cancer prognosis (Gabelli et al., 2010). The ideal "hot-spot" mutant to target with anti-cancer agents would have both an

activating effect on the protein and exploitable conformational changes when compared to its wild-type counterpart.

Gain-of- function mutations in cancer genes predominantly occur at specific protein residues or active domains. An example is PIK3CA in which most mutations are predominantly found in the kinase and helical domains of the protein (Karakas, Bachman, & Park, 2006). However, remains to be clarified how it can be used to nominate drivers, not just gain-of-function cancer genes (Tamborero et al., 2013).

Breast cancer

In this study, we focus in breast cancer, because is the second leading cause of cancer death in women, 1 in 38 (about 2.6%). Only lung cancer kills more women each year. And the most common cancer in American women, except for skin cancer, the average risk for developing it sometime in her life is about 12% (1 in 8) (American Cancer Society, 2018).

It is therefore important to identify markers that can predict tumor aggressiveness and predict the response to the selected therapy at the same time than new functional targets can be identified (Børresen-Dale, 2003).

Death rates from female breast cancer dropped 39% from 1989 to 2015 (Fig. 2) and since 2007 have been steady in women younger than 50, but have continued to decrease in older women (American Cancer Society, 2018).

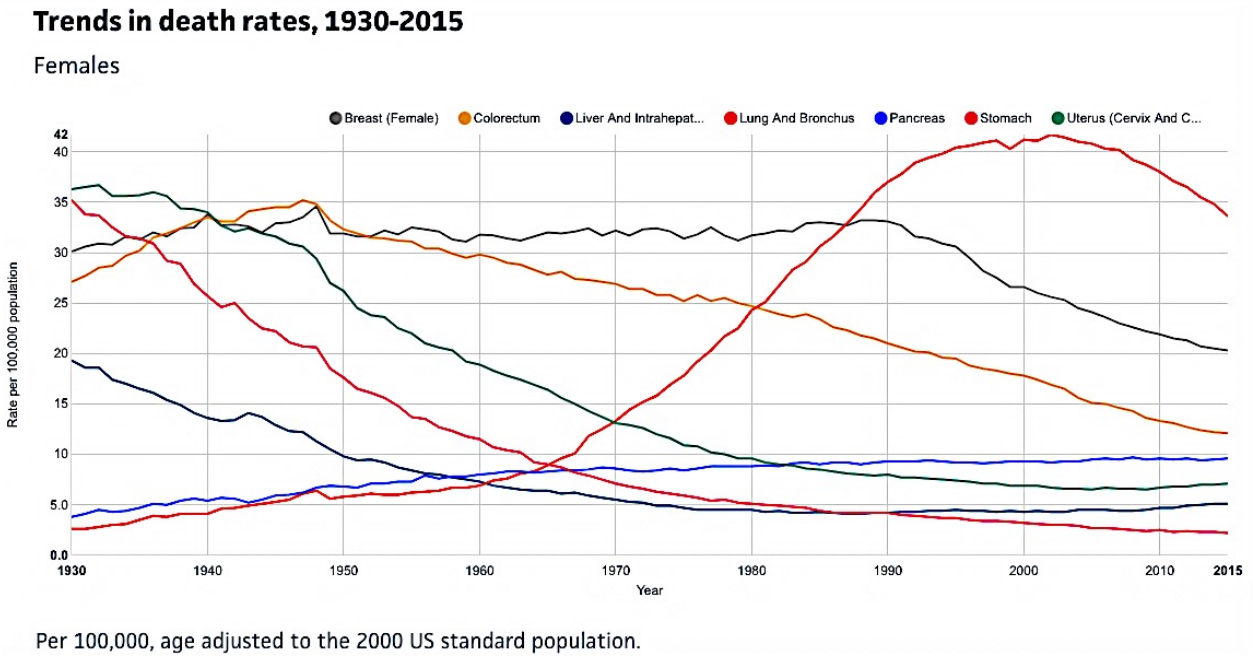


Figure 2. Trends in death rates from 1930 to 2015. Breast cancer is the second cause of cancer death in women in United States. *Data sources: National Center for Health Statistics (NCHS), Centers for Disease Control and Prevention, 2017. Picture taken from American Cancer Society webpage.*

These decreases are believed to be the result of earlier diagnostic, as well as improved treatments, due to the increase on studies that help to understand this disease.

Mutations in DNA can cause normal breast cells to become cancer. Some of them are inherited but most of them are acquired, like lifestyle-related risk factors (what you eat and how much you exercise). So these changes, somatic mutations, take place in breast cells during a person's life, which can increase the chance of developing breast cancer. But it's not yet known exactly how some of these risk factors cause normal cells to become cancer.

Gene *TP53*

Most frequent mutations, in approximately 30% of the tumors, are located in the *TP53* gene (Børresen-Dale, 2003). The frequency of mutations in this gene reported in breast tumors ranges from 15 to 71%.

The phosphoprotein is a transcription factor, which regulates apoptosis, genomic stability, and angiogenesis. Functional loss of p53 can lead to defective DNA replication and malignant transformation (Levine, 1997).

The missense mutations usually are between exons 4–10, encoding the DNA-binding and oligomerization domain; specifically, along the domain required for interactions with FBX042, HIPK1, AXIN1, the DNA major groove, and the domain that contains the nuclear export signal (Bai et al., 2014).

Genes *BRCA1/2*

BRCA1 (BReast CAncer) and *BRCA2* human genes encode tumor suppressor proteins, involved in the mechanisms of DNA repair. These genes play a big role in preventing breast cancer (National Breast Cancer Foundation, 2018). When mutations in either of these genes affect the protein products, DNA damage may not be repaired properly. As a result, cells are more likely to develop additional genetic alterations that can lead to cancer.

Mutations of *BRCA1* or *BRCA2* genes occur in 0.25% (about 1 in 400) of the population (National Breast Cancer Foundation, 2018).

PI3K-Akt-mTOR signaling pathway

In 2004, Campbell *et al.* sequenced all of the 20 coding exons of PIK3CA from primary tumor samples of breast cancer and suggested that together with other studies (Broderick et al., 2004; Saal et al., 2005), the PI3K-Akt-mTOR pathway plays a central

role in breast tumorigenesis and implies that this pathway may be an interesting target for the development of novel therapies for cancer (I. G. Campbell et al., 2004).

A better understanding of the pathology of **PI3K-Akt-mTOR pathway´s signaling** in tumors could provide novel therapeutic targets (Parsons, 2004).

The **PI3K-Akt-mTOR signaling pathway** (Fig. 3) is a key regulator of cellular processes involved in cell growth, proliferation, motility, survival, and apoptosis (Thorpe, Yuzugullu, & Zhao, 2015). Alterations of this pathway affect the survival and proliferation of tumor cells in many human cancers (Porta, Paglino, & Mosca, 2014).

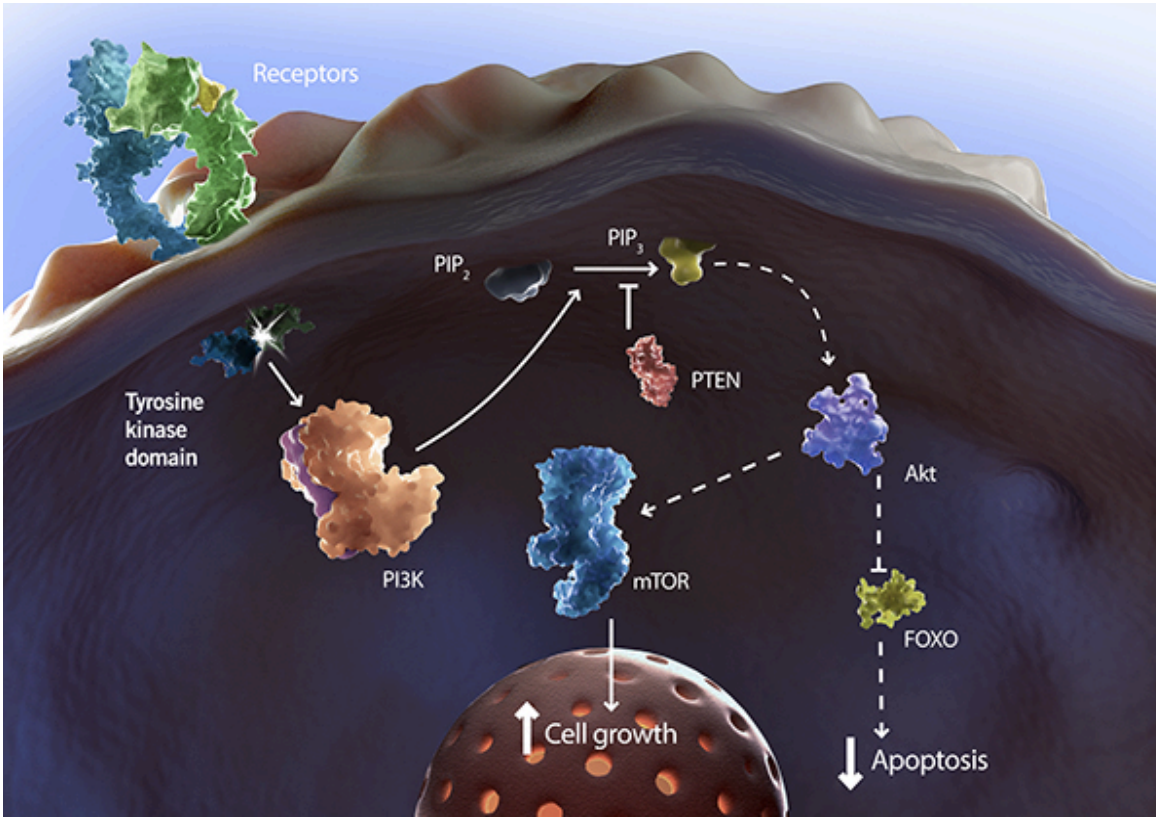


Figure 3. Growth factor stimulation of receptor tyrosine kinases triggers PI3K activation, downstream activation of phosphoinositide-dependent kinase 1 (PDK1) and AKT, and, subsequently, mTOR complex 1 (mTORC1), which promotes cell growth and protein synthesis (2). The pathway can be activated by RTKs, G protein-coupled receptors (GPCRs) or by oncogenic proteins such as RAS (Liu et al., 2009). The tumor suppressor phosphatase and tensin homolog (PTEN) is a key negative regulator of the PI3K pathway (LoRusso, 2016). Picture taken from www.biooncology.com (Genentech website).

Phosphatidylinositol 3-kinases (**PI3Ks**) are heterodimers, composed of catalytic and regulatory subunits, that are activated by growth factor–receptor tyrosine kinases (**RTK**s) (Lewis C. Cantley, 2002; Vanhaesebroeck & Waterfield, 1999). RTKs recruit the catalytic

subunit to the membrane, allowing the activated PI3Ks catalytic subunit to phosphorylate the second messenger phosphatidylinositol 4,5-bisphosphate [PI(4,5) P2 or **PIP_2**], converting it to phosphatidylinositol 3,4,5-trisphosphate (**PIP_3**) (Lewis C. Cantley, 2002; Vanhaesebroeck & Waterfield, 1999), which levels are tightly regulated by the action of phosphatases, such as the phosphatase and tensin homolog (**PTEN**) (Maehama & Dixon, 1999) (Fig. 3).

Once at the membrane, PIP_3 recruits the serine-threonine protein kinases **Akt** (also called protein kinase B) and phosphoinositide-dependent kinase 1 (**PDK1**) to the membrane. PDK1 consequently phosphorylates and activates Akt (Huang et al., 2007; Ikenoue et al., 2005; Ruggero & Sonenberg, 2005). A key protein in the pathway is **mTOR**, that acts both upstream and downstream of Akt, that is active in 2 different multiprotein complexes, rapamycin complex (TORC) 1 and 2. In turn, activate numerous downstream pathways involved in cell proliferation, survival, motility and growth (Lewis C. Cantley, 2002; Carson et al., 2008; Kang, Bader, & Vogt, 2005; Slomovitz & Coleman, 2012; Vivanco & Sawyers, 2002) (Fig. 3).

Phosphatidylinositol 3-kinase (**PI3K**), **Akt** (a serine/threonine kinase also known as PKB), and mammalian target of rapamycin (**mTOR**) are 3 major junctions in the pathway, and are typically activated by upstream signaling of tyrosine kinases and other receptor molecules such as hormones (Ruggero & Sonenberg, 2005).

Aberrant activation of the PI3K-Akt-mTOR pathway is implicated in many human tumors (Carson et al., 2008), including breast cancer (Ikenoue et al., 2005). Activation of this pathway can be induced by mutations of the PIK3CA gene (encoding the catalytic α-subunit of PI3K), inactivation of the PTEN tumor suppressor gene, Akt overexpression and gene amplification. The 3 key components of this pathway are PI3K, Akt, or mTOR, associated with tumor progression and resistance to cancer therapies when are molecularly altered (Chen et al., 2017)(Ikenoue et al., 2005; Myers & Cantley, 2010).

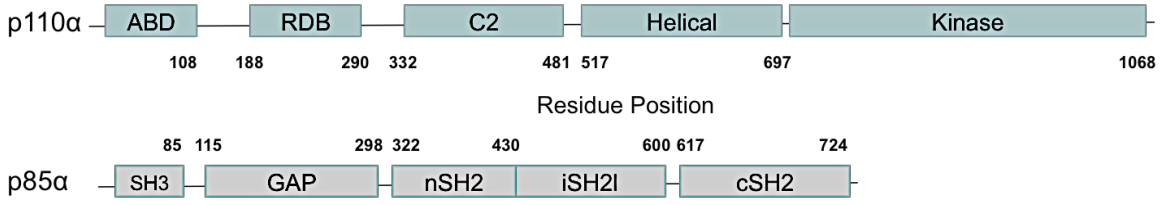
Gene *PIK3CA*

Phosphatidylinositol 3-kinases (**PI3Ks**) are lipid kinases divided in 3 classes, each with its own substrate specificity, tissue distribution, and mechanism of action (Cantrell, 2001; Fruman, 1998; Katso, Okkenhaug, Ahmadi, Timms, & Waterfield, 2001). Class I PI3Ks can be further subdivided into two groups (IA and IB) based on their structure and

mode of activation (Liu, Cheng, Roberts, & Zhao, 2009; Wymann, Zvelebil, & Laffargue, 2003). *PIK3CA*, one of the two most frequently mutated oncogenes in human tumors (Gabelli et al., 2010; Liu et al., 2009; Thorpe et al., 2015), codes for p110α, the catalytic subunit of PI3Kα (p110α/p85, fig. 4b), that belongs to Class I A heterodimeric PI3Ks (Bader, Kang, Zhao, & Vogt, 2005) (Huang et al., 2007).

The p110α (catalytic subunit of PI3Kα) has five domains (Fig. 4a): an N-terminal domain called ABD (adaptorbinding domain) that binds to p85α, a Ras binding domain (RBD), a domain called C2 that has been proposed to bind to cellular membranes, a helical domain of unknown function, and a kinase catalytic domain (Lewis C. Cantley, 2002; Katso et al., 2001; Vanhaesebroeck & Waterfield, 1999; Vivanco & Sawyers, 2002).

a



b

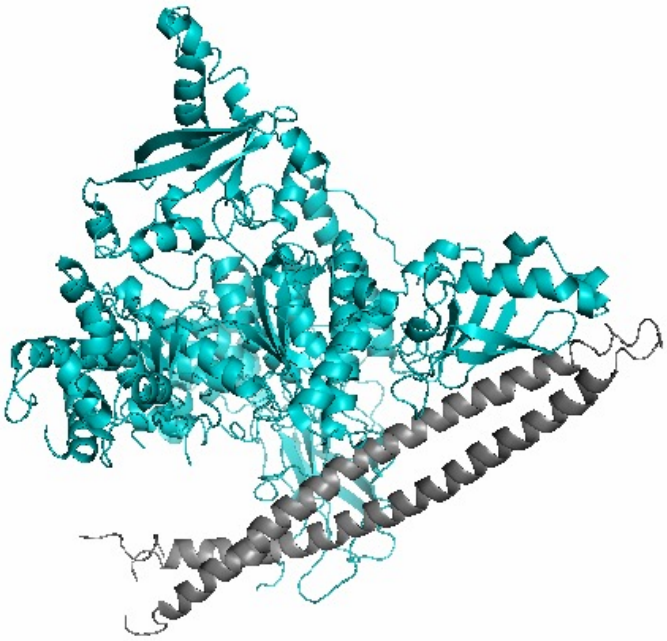


Figure 4. a) PI3K protein is a heterodimer: p110α (encoded in *PIK3CA*) and p85α, representation of the domains in each subunit. **b)** PI3K protein 3D structure, in blue catalytic subunit and in grey regulatory subunit.

The p85α polypeptide (regulatory subunit of PI3Kα) also has five known domains (Fig. 4a) including two SH2 domains (the N-terminal nSH2 and C-terminal cSH2) separated by an inter-SH2 (iSH2) domain that binds to the catalytic subunit.

In its basal state, p85 regulatory subunit stabilizes and maintains p110 catalytic subunits of PI3K in a quiescent state until activated by receptor tyrosine kinases (Cheung et al., 2014; Cuevas et al., 2001; Yu, Wjasow, & Backer, 1998). When appropriate cellular stimuli are present, the nSH2 and cSH2 domains bind phosphorylated tyrosines (Tyr-X-X-Met motifs) found in activated receptors and adaptor proteins, and this phosphotyrosine binding activates the p110 catalytic subunit without releasing p85 from p110. PI3K phosphorylates PIP_2 and converts it to PIP_3 (Ikenoue et al., 2005).

This structure suggests that p85α regulates p110α activity through a helix aK12–mediated conformational change of the activation loop (Huang et al., 2007) (Yu et al., 1998).

Most of the reported mutations in PIK3CA cluster in conserved regions within the region coding for the helical and kinase domains of p110 a (Huang et al., 2007). As these mutations constitutively activate its kinase activity, **the enzyme appears to be an ideal target for drug development**. Progress in this area of drug development would be facilitated by knowledge of the structure of the p110a/p85a complex.

Gene *AKT1*

Akt or protein kinase B, a serine-threonine kinase, is an important downstream effector of PI3K (Wan, Harkavy, Shen, Grohar, & Helman, 2007), and one of the most frequently activated protein kinases in human cancers (Wan et al., 2007). There are 3 highly homologous Akt isoforms (Akt 1, 2, and 3) that are encoded by separate genes and share over 80% amino acid sequence identity in mammalian cells, with a similar structure: an N-terminal PH domain, a central serine-threonine catalytic domain, and a small C-terminal regulatory domain (Hay, 2005; Liu et al., 2009).

Binding of Akt to PIP_3 occurs at cellular membranes (Kang et al., 2005), resulting in a conformational change in AKT, exposing two critical amino acid residues for phosphorylation by PDK1 and PDK2 (Liu et al., 2009).

In cancer, Akt activity is frequently elevated due to oncogenic growth factors, angiogenic factors, cytokines, steroid hormones (estrogen and androgen), and genetic alterations, including mutations and/or amplifications of the *AKT1*, *AKT2*, and *AKT3* genes; loss of function of the *PTEN* tumor-suppressor gene; and mutations of the *PIK3CA* gene (Altomare & Testa, 2005; Cheng, Lindsley, Cheng, Yang, & Nicosia,

2005; Malanga et al., 2008) . Hyperactivation of Akt may induce cell growth and proliferation, and contribute to apoptotic resistance (Wan et al., 2007).

Gene *PTEN*

PTEN is the most important negative regulator of the PI3K-Akt-mTOR pathway by dephosphorylating PIP_3 second messenger (L C Cantley & Neel, 1999; Cully, You, Levine, & Mak, 2006). Can be divided into three domains: a phosphatase domain (1–185), a C2 domain (186–352), and a tail domain (353–403) (Lee et al., 1999). The phosphatase and C2 domains are required for efficient membrane binding. Mutations in the phosphatase motif inhibit PIP_3 catalysis. The tail domain is an important region for negative regulation of PTEN. Deletion of this region activates PTEN's ability to inhibit Akt (Parsons, 2004).

Lack of PTEN in a cell favor tumorigenesis, leading to increase PIP_3 and Akt kinase activity, thus reduce apoptosis, increase proliferation and alter migration (Parsons, 2004).

Gene *mTOR*

Another serine-threonine kinase, mTOR (mammalian target of rapamycin) that exists in 2 distinct intracellular complexes: mTORC 1 and 2, plays an important role in the regulation of cell growth and proliferation by promoting protein synthesis (Wullschleger, Loewith, & Hall, 2006).

Activation of mTORC1 is achieved through PI3K and Akt (Pópulo, Lopes, & Soares, 2012). Aberrant activation of mTOR has been implicated in a variety of malignancies, including breast cancer (Advani, 2010).

Previous studies

One of the most known methods to detect positive selection is based on frequency. Therefore, a commonly used method is to consider mutations that occur with higher frequency than a background mutation rate (Dees et al., 2012; Getz et al., 2007).

Tamborero et al., (2013) developed a method to identify cancer related genes by grouping together residues with a significant rate of mutations that are close in the primary sequence of the protein above the background model (Fig. 5). The background model was generated considering coding-silent mutations based on the evidences of a nonrandom mutation processes along the genome (Amos, 2010).

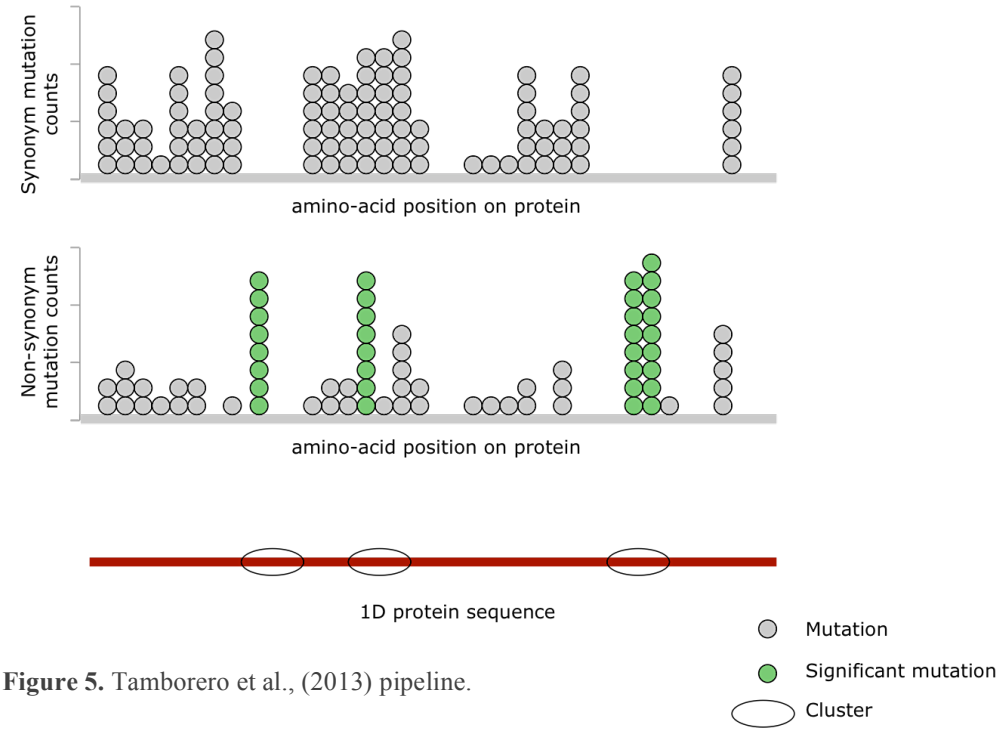


Figure 5. Tamborero et al., (2013) pipeline.

Recently, Gao et al., (2017) identified genomic mutations affecting residues located in 3-dimensional proximity of protein structures by comparing the mutation frequency against a random background (Fig. 6).

The first method (Fig. 5) used gene sequences, considering proteins as single strands, and omitted that distant genomic regions might be close in the 3D space when the protein folds. And the second method (Fig. 6) assumed a homogeneous mutation probability across the whole genome, which is likely an oversimplification that may introduces a bias in the expected mutation rates (Amos, 2010).

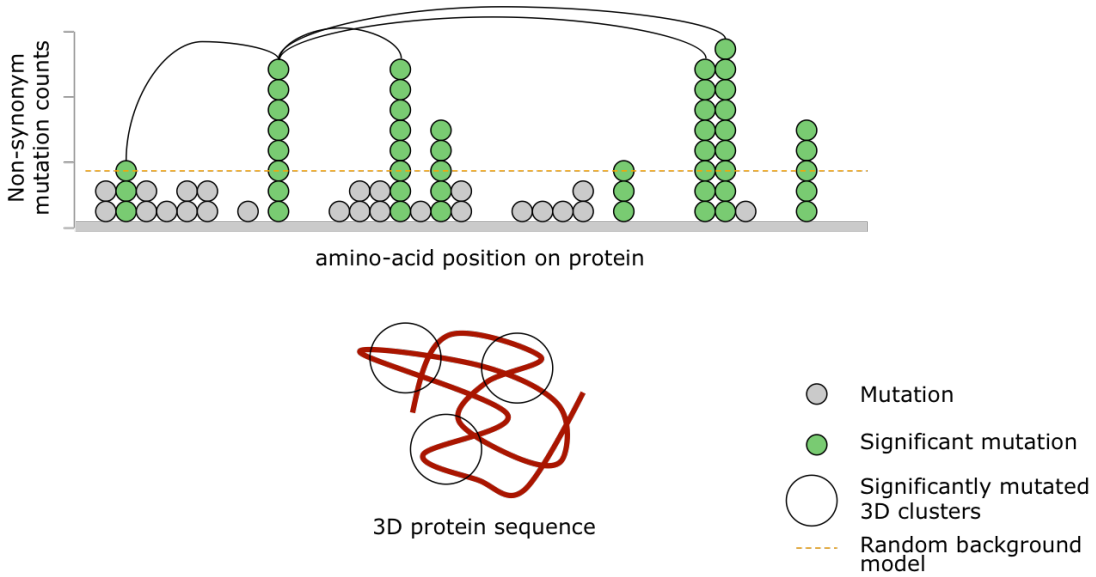


Figure 6. Gao et al., (2017) pipeline.

Both problems were considered in this study for the development of the algorithm. This method identifies associated with BRCA-mutated breast cancer using coding-silent mutation frequency as a background. Furthermore, the method identified structural and catalytic roles of 3D protein clusters within relevant biological pathways in breast cancer.

The method developed in this study (Fig. 7) considered that a 3D protein cluster is significant when the residues within it have a higher non-synonymous mutation rate as compared to the background mutation rate.

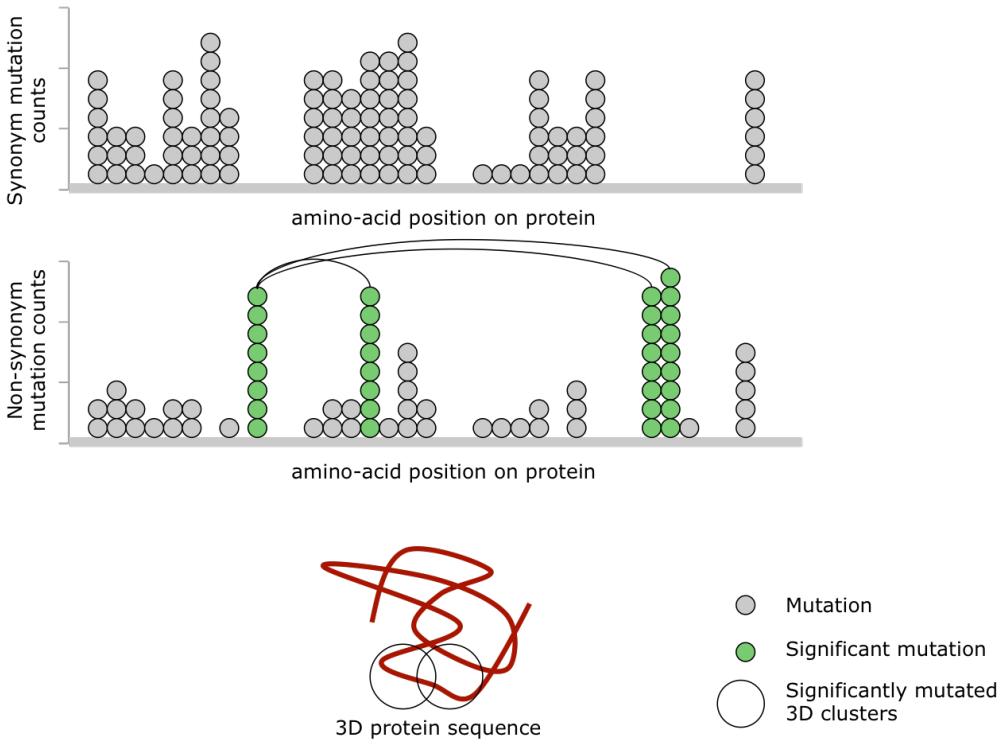


Figure 7. My Master´s Thesis pipeline

Material and methods

For the development of the project, as input, this tool requires files stating the position of each mutation within the protein sequence to group those ones that are in spatial proximity and generate 3D clusters.

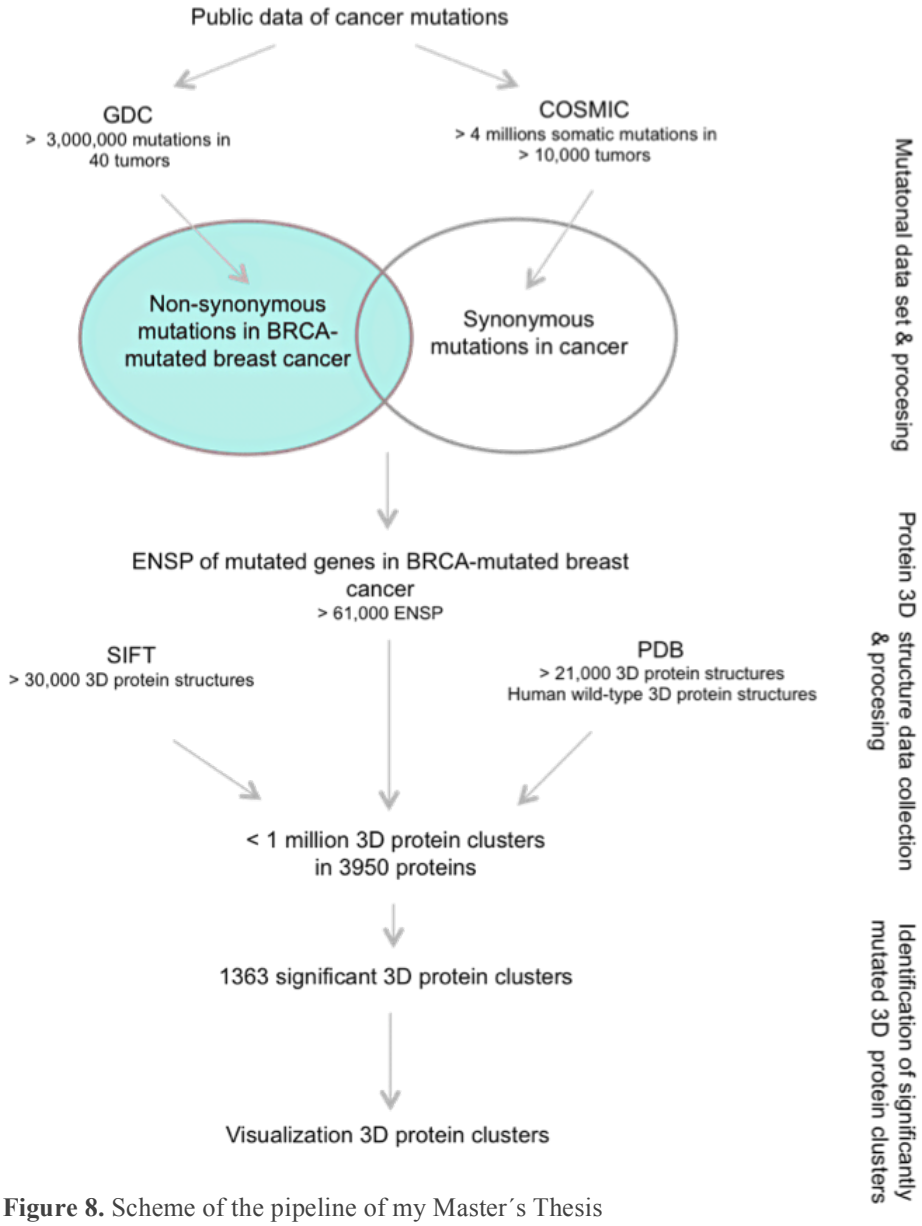


Figure 8. Scheme of the pipeline of my Master´s Thesis

The method was applied only to mutational data from BRCA-mutated breast cancer patients. The two data sets used (Fig. 8), were obtained from publicly available sources: The Cancer Genome Atlas (TCGA) and Catalogue of Somatic Mutations in Cancer (COSMIC). In both of them the genomic coordinates of variants were standardized to the human reference assembly GRCh38.

From GDC (Genomic Data Commons) Data Portal, single-nucleotide protein affecting mutations were collected from BRCA-mutated breast cancer patients open access dataset from The Cancer Genome Atlas (TCGA) (Fig. 9). The data was divided in 986 VCF files, every one corresponding to the mutations in each patient, grouped in one MAF file with

120,988 mutation counts, 61582 were non-synonymous (cancer related), corresponding to 15587 genes from 986 patients (Annex II).

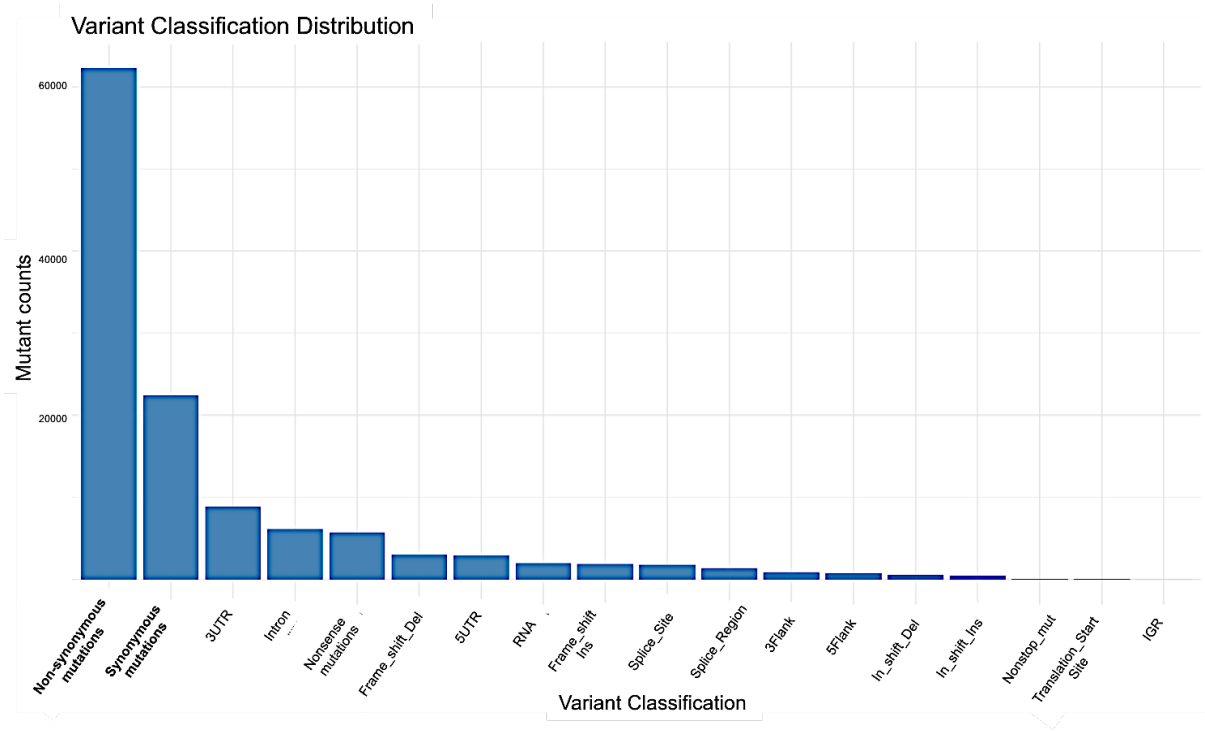


Figure 9. Variant Classification Distribution of BRCA-mutated breast cancer MAF file

The second dataset was downloaded from the Catalogue of Somatic Mutations in Cancer (COSMIC, http://www.sanger.ac.uk/cosmic) v78 (Forbes et al., 2016). This database contains over 4 million coding mutations across all human cancer disease types, of which more than 700,000 are unique coding-silent mutations from 16,460 patients.

Coding-silent mutations in cancer patients were retrieved to create the background model (Annex III). Coding-silent mutations are assumed not to be under positive selection and thus may reflect the baseline tendency of somatic mutations to be clustered.

Identification of 3D protein clusters

Filtering

These two filtered datasets by the corresponding mutations were merged, by selecting only the synonymous mutations located in those genes with non-synonymous mutations. A unique list (Annex II - Ensembl_ID.csv) with the ENSP (Ensembl protein identifier) of each protein was obtained. Next, the mutational data was integrated with structural information to detect the clusters significantly enriched of cancer related mutations.

Selection of the PDB files

Protein structures (PDB files) with sequence similarity to the Uniprot entries, of 90% or above, were included. Sequence similarities were retrieved from the Structure Integration with Function, Taxonomy and Sequences (SIFTS) resource (Velankar et al., 2013). Only PDB files from human wild-type structures with sequence identity of 90% to proteins annotated in UniProt were selected (Annex VI).

From 61,582 ENSPs, 61,136 were found in UniProt (Bateman et al., 2015) but only 4289 of them had a 3D structure in the Research Collaboratory for Structural Bioinformatics (RCSB) Protein Data Bank (PDB, http://www.rcsb.org/) (Berman et al., 2000), that corresponded to 3,950 proteins (Annexes IV and V).

For this purpose, structures of the proteins encoded by cancer related genes were downloaded and the atomic position of the residues in the proteins was used to generate the 3D clusters of the protein structures.

Identification of 3D protein clusters

Once the selected protein structures were downloaded (Annex VII), contact map of residues (Gao et al., 2017) were calculated (Annexes VIII and IX). In this way, 3D protein clusters were defined by a central alfa-carbon of a protein residue and all the alfa-carbon of residues within 15 angstroms (Å). All residues were used as centers of clusters and the 3D cluster score was computed separately for each cluster. Each residue can appear in more than one cluster. We obtained 1,125,682 3D protein clusters in total.

Computation of cluster mutation frequencies

The genomic position of each mutation and the corresponding mutant count were mapped and assigned to the sequence position of the encoded amino-acid (Annexes IX and X), thus to each 3D protein cluster.

Then the non-synonym (NS) and synonym (S) mutation frequencies (1, 2) were computed with the mutant counts as shown below:

$$NS\ frequency = NS\ mutant\ counts * cases * length \quad (1)$$

$$S\ frequency = S\ mutant\ counts * cases * length \quad (2)$$

where *NS mutant* counts represents the number of times that a cancer related (non-synonymous) mutation appears in a position in the BRCA dataset. The same way, *S mutant counts* represents the number of times that a coding-silent (synonymous) mutation appears in a position in the COSMIC dataset. *Cases* is the number of patients in the database (16,460 for COSMIC dataset and 10,188 for GDC dataset), even there are 986 BRCA-mutated breast cancer patients, we use the total number of cases in the dataset. *Length* is the number of residues in the cluster. To do comparable the frequencies, the length of the cluster is multiplied to the length of the cluster, because is not the same two mutations in a cluster of 15 residues than in a cluster of 30 residues.

Computation of the 3 D protein score

The 3D cluster score is computed (Annex XI) as (3):

$$3D\ cluster\ score = \frac{NS\ mutant\ counts * cases * \cancel{length}}{S\ mutant\ counts * cases * \cancel{length}} = \frac{NS\ frequency}{S\ frequency} \qquad (3)$$

The 3D cluster score is directly proportional to the frequency of non-synonym mutations of the residues in the cluster and inversely proportional to the frequency of synonym mutations in the cluster. The number of residues in each cluster does not affect the 3D cluster score since the frequency of synonyms and non-synonyms mutation are computed in the same set of residues.

There are two reasons why the 3D cluster score was computed as simple ratio: i) the distribution of the synonym mutation frequencies is not normal therefore a Z-score cannot be applied, ii) genomic regions with no mutations annotated generate frequency of value 0 that are called structural zeroes. The presence of structural zeroes limits the application of statistical methods.

The presence of structural zeroes does not allow the computation of 3D cluster scores. Therefore, a pseudo-count (10^{-10}) was added to each mutant count.

The clusters were ranked based on the 3D cluster score after $\log_2$ transformation in order to make the frequencies comparable (same scale).

All the algorithms were written in Python or R languages (Annexes II-XI). The entire bioinformatics pipeline was executed in Computerome (Danish National Supercomputer for Life Science).

Visualization of 3D protein clusters

The 3D structures of proteins with a high number of significant 3D cluster scores were further analyzed with PyMOL, a free cross-platform molecular graphics system.

PyMOL is very useful when working with proteins, because it allows localizing the aminoacids in a 3D space. Therefore, PyMOL allows the visualization of non-synonymous mutations of significant 3D protein clusters within the 3D protein structure. This gives information about how the residues interact and how the mutations can affect the structure and also to the functionality of a protein (see Results).

Results

Analysis of the ratio between non-synonymous and synonymous mutation frequencies is a good measure of positive selection during tumor progression, as synonymous alterations are unlikely to exert a growth advantage.

Somatic mutations were collected and merged from two different databases. The dataset from TCGA with mutations of BRCA-mutated breast cancer patients, contained 120,988 mutation counts, 61,582 of them were non-synonymous (cancer related) located in 15,587 genes from 986 patients. The second dataset was from COSMIC, with more than 4 million coding mutations across all human cancer disease types, of which more than 700,000 mutations are unique synonymous alterations from 16,460 patients.

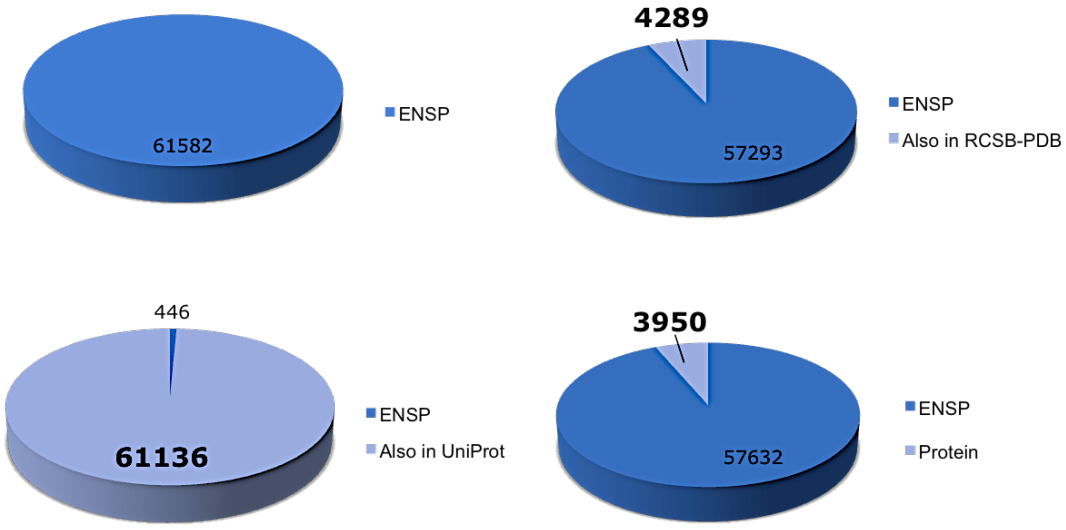


Figure 10. Representation of the difference between the number of ENSP founded in Ensembl and the number of PDB files founded in RCSB**.** Due to a lack of complete protein structure data, some of the PDB files included only individual protein domains.

These datasets contains the position of each mutation within the protein sequence to group those ones that are in spatial proximity and generate 3D clusters.

From 61,582 ENSPs in common between the two datasets, 61,136 were found in UniProt (Bateman et al., 2015) but only 4,289 of them had a 3D structure in the Research Collaboratory for Structural Bioinformatics (RCSB) Protein Data Bank (PDB, http://www.rcsb.org/) (Berman et al., 2000), that corresponded to 3,950 proteins (Fig. 10).

The difference between the number of ENSP that we had at the beginning (Fig. 10) and the reduced number of 3D structures corresponding to the proteins of some of them, is because we only selected those PDB files from human wild-type structures with 90% of sequence identity to proteins annotated in UniProt (see Materials and methods).

When the method developed in this study was applied to BRCA-mutated breast cancer patients, more than 1 million 3D protein clusters (1,125,682) located in 3,950 proteins were generated. Then, these clusters were ranked based on the $\log_2$ 3D cluster scores as shown in the graph below (Fig. 11).

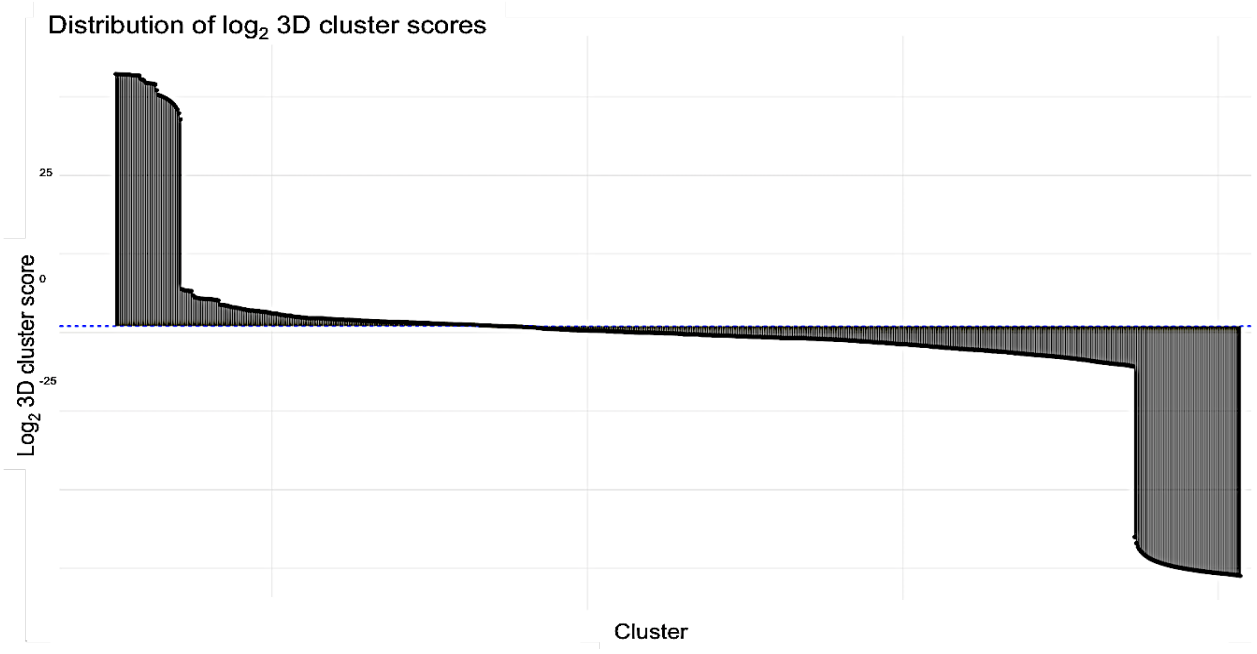


Figure 11. The 3D cluster score is a ratio. The $\log_2$ of it equals 1 when the frequencies for synonymous and non-synonymous mutations in a cluster are the same. Values lower than 1, right side of the graph, are those spatial clusters with higher synonymous mutation frequencies. And values higher than one, left side of the graph, correspond to the spatial clusters with higher non-synonymous mutation frequencies.

Due to the pseudo-counts added to each mutant count, the shape of the graph was expected. And it highlighted those 3D protein clusters with only one kind of alterations,

synonymous or non-synonymous mutations, located at the extremes of the distribution (Fig. 11).

In this analysis, we focused on the left side of the graph (Fig. 11), where we expected to find the 3D protein clusters that contained potential functional targets. The 3D protein clusters with a value higher than 1, indicate a positive selection, mutations that can benefit the cell proliferation and tumorigenesis development. However, **not all this alterations in cancer genes are driver alterations**. Furthermore, **not all driver alterations in a cancer gene have the same functional impact**.

We expected to find three types of clusters: i) with mutations close in the sequence, ii) containing mutations far in the sequence but within the same domain and iii) with grouped mutations that are localized in different domains, thus far in the sequence. This last one is the most interesting because mutations that are not close in 1D protein sequence cannot be easily related. Identifying interactions of non-synonymous mutations, localized in different domains but close in the 3D protein space, can contribute to identify new targets, develop new therapies and consequently maximize the therapeutic benefit (Fig.7).

Even 1 million of 3D protein clusters were computed, only 1,363 with values higher than 5, were considered significant. These clusters were localized in 27 genes and with a 3D structural representation in 28 PDB files. Two of these 3D structures included only individual protein domains, not the complete protein structure.

In this BRCA-mutated breast cancer dataset, many of the significant 3D cluster scores were identified in two well-characterized cancer genes: *PIK3CA* (Fig. 12) and *TP53* (Annex I). Other significant 3D protein clusters were localized in *AKT1* and *PTEN* genes (Annex I). All of them related with tumorigenesis development and cancer implications and involved in PI3K-Akt-mTOR pathway.

PIK3CA

The oncogenic potential of the PI3K pathway is explained by two key observations. First, alterations in the PI3K/AKT/mTOR pathway can induce cell line transformation and tumor formation in transgenic mice (Carver et al., 2011; Engelman et al., 2008). Second, PI3K signaling activation frequently occurs following multiple molecular alterations in other components of the pathway downstream of PI3K, such as mutations in *PIK3CA, AKT1* and *PTEN* genes.

Most of the significant 3D protein clusters located in the left side of the graph corresponded to the gene *PIK3CA* (Fig. 12). When computing the ratio between non-synonymous to synonymous mutations, in this gene, there is a prevalence for non-synonymous changes above the background model.

Non-synonymous mutations in *PIK3CA* gene have been reported in many human cancer types (Karakas et al., 2006), and in breast cancer, most mutations occur in this gene. In 2016, LoRusso published that **approximately 20% to 50% of breast cancers exhibit PIK3CA mutations.** Three frequent hotspot mutations (Fig. 13) within the helical (E545K and E542K in exon 9) and kinase domains (H1047R in exon 20) result in constitutive p110a (catalytic domain of PI3K) activity (Karakas et al., 2006; LoRusso, 2016).

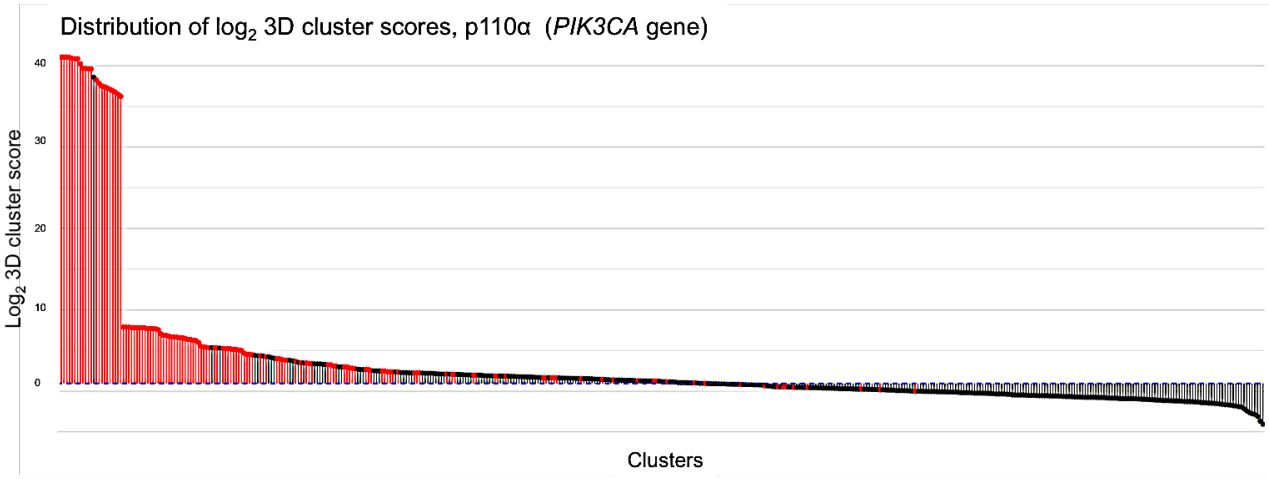


Figure 12. Distribution of log_2 3D cluster scores, highlighted in red the 3D cluster scores within PIK3CA gene

In addition to these three hot spots, there are several cancer-specific mutations that have been shown to result in enhanced enzymatic activity of PI3K in vitro and in vivo (Huang et al., 2007; Ikenoue et al., 2005; Kang et al., 2005; Samuels et al., 2004):

- ABD domain mutations: (residues R38 and R88) the structure of the p110a/niSH2 reveals that both Arg38 and Arg88 are located at a contact between the ABD and the kinase domains.
- Mutations in the C2 domain: (residues N345 and E453) mutations in both residues may disrupt the interaction of the C2 domain with iSH2 and alter the regulatory effect of p85 on p110a.
- Helical domain mutations: (residues E**542**, E**545** (hotspots) and Q546) mutations in these residues, located on an exposed region, can cause a charge reversal. Interaction

with residues of nSH2, p85 domain, alter the activity of the catalytic subunit (Miled et al., 2007; Yu et al., 1998). Mutations may modify the orientation of nSH2 with respect to the helical and the kinase domains.

- Kinase domain mutations: (residues H**1047** (hotspot) and M1043) residue H1047 is a hot spot for somatic mutations in cancer. Mutations in this residue can change the interaction between the activation loop and the substrates. Several studies have shown support for the hypothesis that activating somatic mutations tend to cluster in protein kinases (Dixit et al., 2009; Greenman et al., 2007; Izarzugaza, Redfern, Orengo, & Valencia, 2009).

In this study, not all of the mutated residues were founded, such as residue 453; or had a low mutation frequency, fewer than 5 mutation counts, such as residues 38, 88 and 379. Furthermore, residues with high mutation frequencies in this dataset, located in positions 420 and 452, are not considered as breast cancer-specific mutations in previous studies. Could be possible functional targets in future studies, because there are located close to residue E453, an already known important cancer-specific mutation.

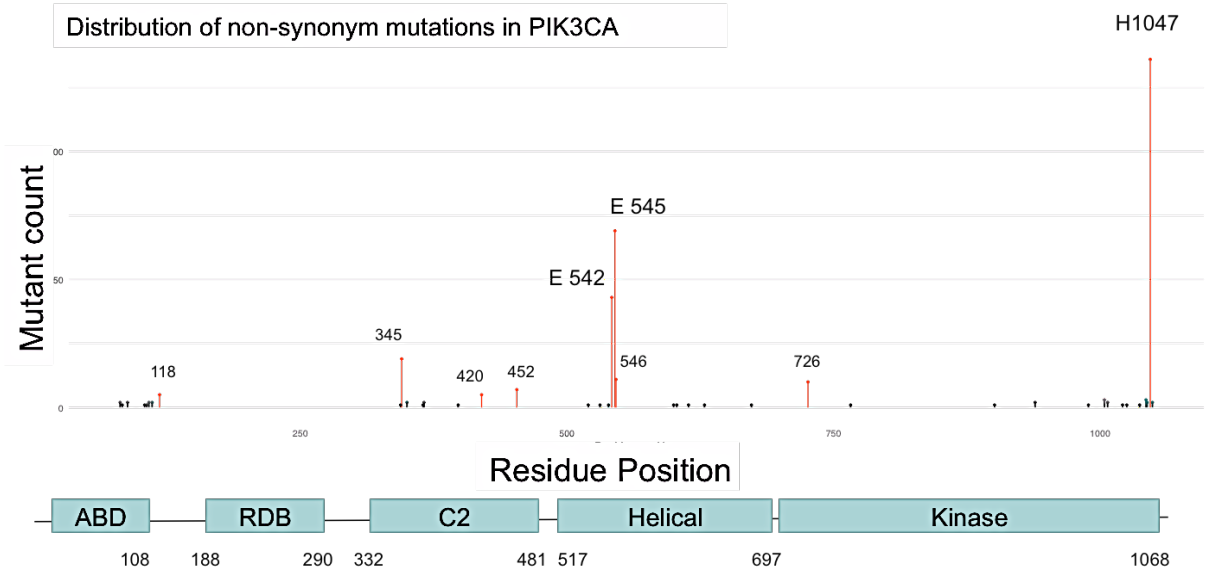


Figure 13. Representation of significant non-synonymous mutations (>5 mutant counts) in *PIK3CA* residues in BRCA-mutated breast cancer patients of TCGA dataset.

As the distribution represents (Fig. 13), the three mutations with higher non-synonymous mutant counts are located in residues: H1047, E545 and E542. In particular, H1047R mutation is located in the kinase domain near the binding region of the activation

loop. Is postulated to **enhance the catalytic activity of the enzyme by locking the loop in the 'on' position**. On the other hand, alterations in E542K and E545K are localized in the helical domain. These mutated residues are clustered into an **exposed region and might interact with the cellular membrane** (Carson et al., 2008). Therefore, these mutations cause increased lipid kinase activity (2-fold increase, Carson et al., 2008) by allowing easier access to the membrane-bound PIP_2 substrate, which is then converted to PIP_3, initiating tumorigenic signaling cascades (Karakas et al., 2006; Mandelker et al., 2009).

The most significant 3D cluster scores of PIK3CA included the three hotspots. As it is expected, mutations in the helical domain, E545 and E542, are clustered together (Fig. 14) due to the proximity in the sequence. But, we also founded that mutations in residues E542 (helical domain) and H1047 (kinase domain), where grouped together (Fig. 15) in the same cluster. This one is very interesting, because contains two hotspots localized far in the sequence, in different domains, but close in the 3D spatial protein structure.

Also, it is important to highlight, even some of these mutations can be identified by their mutation frequency in a single position, and others are less common, below 5 mutant counts. These rare mutations commonly located in 3D proximity to hotspots or to a known and common mutation in the same protein, can be considered as possible driver events.

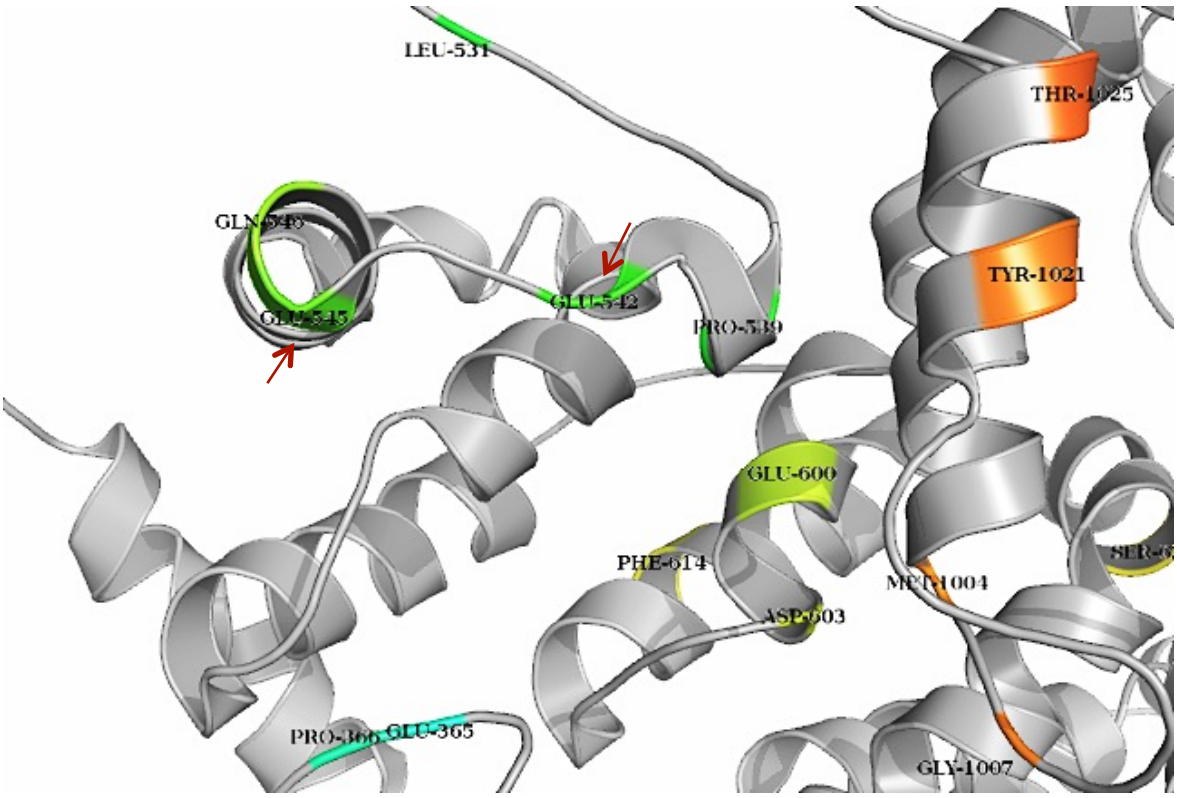


Figure 14. 3D visualization of two of the hotspots, residues E542 and E545, in PIK3CA gene highlighted with red arrows

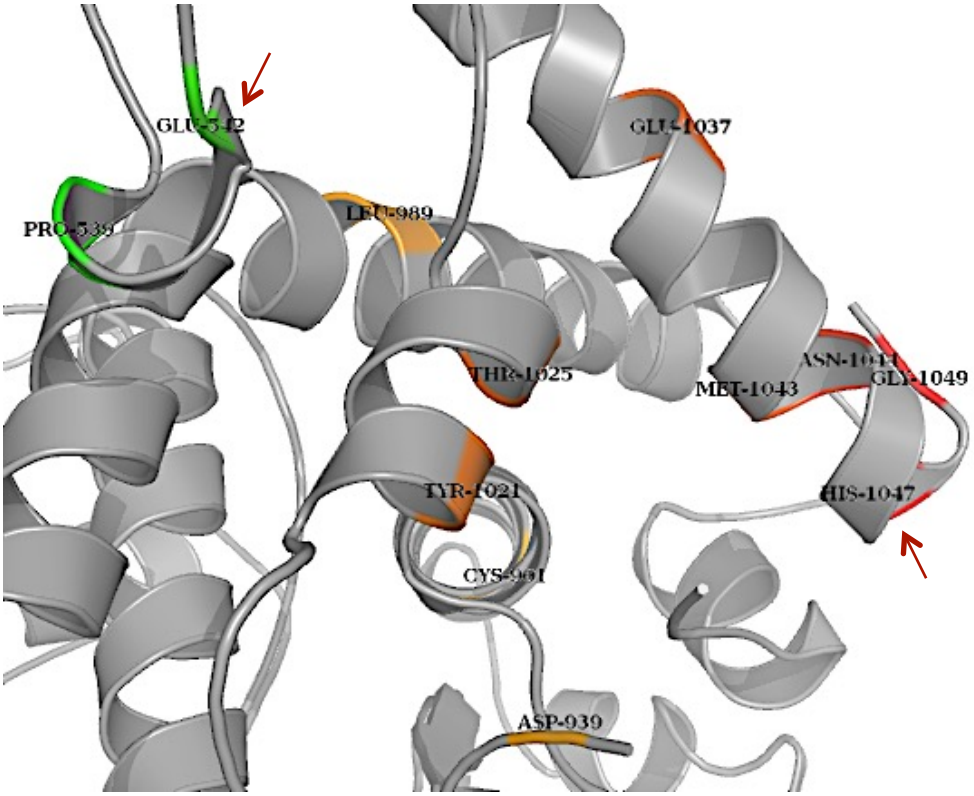


Figure 15. 3D visualization of two of the hotspots, residues E542 and H1047, in PIK3CA gene highlighted with red arrows

Discussion

Importance of this study

The acquisition of somatic mutations can induce cancer by **dysregulating the delicate mechanisms controlling balance between** proliferation and apoptosis. **Genomic alterations can be classified in** driver and passenger mutations.

Most of the driver mutations have unknown functional impact on protein structure and function. Furthermore, **not all driver alterations in a cancer gene have the same functional impact**. Identification of driver mutations among the passenger mutations in patient genomes it is necessary for effective development and use of targeted therapies.

The use of high-throughput sequencing technologies facilitated the discovery of cancer related mutations in case and control studies. The analysis of different tumor types facilitates the identification of recurrent mutations and the functional pathways involved in tumor development.

Many of the somatic mutations can be identified by their frequencies in a single position, but others are less common. This less frequent mutations commonly located in 3D proximity to hotspots or to a known and common mutation in the same protein, have the possibility that are also driver events.

This method enabled the identification of significant 3D protein clusters containing non-synonym mutations that are relatively far away in linear space but relatively close together in 3D space. The results gave useful information about interactions between residues or domains. The identification of affected pathways can help to understand better the effect of non-synonymous mutations in the tertiary structure of proteins and to identify new targets, develop new therapies and consequently maximize the therapeutic benefit.

Limitations

While our approach can identify interesting mutations grouped in 3D protein clusters as targets, the method is still limited by the **lack of complete protein structure data for many genes**. For the 61,582 genes with synonym and non-synonym mutations in our dataset, we were able to align 61,136 of them to one or more protein structures. But only analyzed 3,950 genes, corresponding to 4,289 PDB files with a protein structure that covered more than 90% of the protein length and with a human wild-type structure. Some of the structures only included individual protein domains, so interaction between different regions cannot be related if them are in different PDB files. **This limits the ability of our algorithm to detect mutated residues grouped in 3D clusters that were not close in sequence, for example, those involved in domain-domain interactions.**

Furthermore, most of the genes do not even have a protein structure, like *BRCA1* or *BRCA2*, important proteins in the BRCA-mutated breast cancer dataset. PDB files of these proteins would have been very interesting to see how non-synonymous mutations are clustered in the 3D space and the interaction between these residues.

Like any statistical method, the power of our approach is also limited by the number of available tumor samples. Fortunately, as more cancer genomic data are generated, additional significant 3D clusters will likely emerge.

Another methodological difference was the distance cutoffs that were used to decide whether two residues are interacting in 3D structures. In this study, the 3D protein clusters were defined within a diameter of 15Å from a central alfa-carbon of a

protein residue, while in Gao *et al*. (2017) study the selected distance was 5Å.

The most important difference between this study and the previous ones is that previous studies considered proteins as single strands, and omitted that distant genomic regions might be close in the 3D space when the protein folds.

Another important difference is that these methods described before assumed that mutation probability is homogeneous across the gene sequence, which is likely an oversimplification that introduces a bias in the analysis.

We identified significant mutated 3D-clusters **above coding-silent mutations as a background**, to determine the significance of our observations. These genomic alterations are assumed not to be under positive selection and thus may reflect the baseline tendency of somatic mutations to be clustered.

A **critical step for the choice of therapy, design of clinical trials, or drug development** is to localize non-synonymous mutations in the 3D space and to identify significant 3D protein clusters containing these cancer-related alterations.

Conclusion

The elucidation of cancer drivers relies on identifying the marks of positive selection that occur during the clonal evolution of tumors. The trend shown by protein-affecting mutations to accumulate predominantly in certain gene regions is a fingerprint that may denote events targeted by the tumorigenesis.

For this reason, we propose to construct the background model using synonymous mutations, which are assumed not to be under positive selection and may thus reflect the baseline mutation clustering of the tumor. This assumption also probably comprises a simplification because some coding-silent mutations could in principle alter processes such as chromatin remodeling or mRNA processing. Nevertheless, apart from this, they are not in general functionally involved in tumorigenesis.

Once the background model was created, non-synonymous mutations were mapped to the protein sequence and grouped together in the 3D space, **taking into account the tertiary conformation of the protein**.

The significant 3D protein clusters where selected and we provided a mechanistic explanation to some of them.

PIK3CA is the gene where most of these 3D clusters where localized. H1047R is an ideal hot-spot mutant to target because has an activating effect on the protein and exploitable conformational changes when compared to its wild-type counterpart.

Initial PI3K-directed drugs in clinical trials, consisting largely of non-isoform-selective pan-PI3K inhibitors, have not provided exciting results. However, recent preclinical studies have demonstrated that different PI3K isoforms play divergent roles in cellular signaling and cancer, suggesting that inhibitors targeting individual isoforms may be able to achieve greater therapeutic efficacy (Thorpe et al., 2015).

Nowadays, treatments with PI3K inhibitors are available. Because the oncogenic PI3K-Akt-mTOR pathway activation is achieved in different redundant ways, these mono-therapies are not always effective. Treatments have been limited requiring adapted strategies in each tumor type (LoRusso, 2016) and PI3K inhibitor combinations generally result in cumulative, nonspecific toxicities.

For future development, it would be very interesting to experimentally validate the potential 3D clusters, which include the main driver and some of the low frequent mutations localized close to them. Also, to apply this method to all type of cancers, to find potential targets common in all of them or specific for each type. These results would implicate **new therapies or personalized treatments, focusing on potential targets on pathways than in individual genes**.

This Master Thesis project allowed me to deal with the main aspects of the bioinformatics research such as: i) use public databases and cancer repositories, ii) parsing of data file, the integration of different type of dataset (protein annotation, structures, sequences and genomic data), iii) use of different programming languages (R, python) in a high performance cluster (Computerome, #182 supercomputer in the world) iv) analysis of big dataset and deal with computational time issues, v) use 3D protein structure visualization tools, vi) identification of the most suitable statistical approaches and vi) biological interpretation of the results.

Furthermore, to complete successfully the project I had to collaborate with different researchers within the work environment.

References

Advani, S. (2010). Targeting mTOR pathway: A new concept in cancer therapy. *Indian Journal of Medical and Paediatric Oncology*, *31*(4), 132. https://doi.org/10.4103/0971-5851.76197

Altomare, D. A., & Testa, J. R. (2005). Perturbations of the AKT signaling pathway in human cancer. *Oncogene*, *24*(50), 7455–7464. https://doi.org/10.1038/sj.onc.1209085

American Cancer Society. (n.d.). Retrieved January 30, 2018, from https://www.cancer.org/cancer/breast-cancer/about/how-does-breast-cancer-form.html#references

Amos, W. (2010). Even small SNP clusters are non-randomly distributed: is this evidence of mutational non-independence? *Proceedings of the Royal Society B: Biological Sciences*, *277*(1686), 1443–1449. https://doi.org/10.1098/rspb.2009.1757

Bader, A. G., Kang, S., Zhao, L., & Vogt, P. K. (2005). Oncogenic PI3K deregulates transcription and translation. *Nature Reviews Cancer*, *5*(12), 921–929. https://doi.org/10.1038/nrc1753

Bai, X., Zhang, E., Ye, H., Nandakumar, V., Wang, Z., Chen, L., … Gao, J. (2014). PIK3CA and TP53 gene mutations in human breast cancer tumors frequently detected by ion torrent DNA sequencing. *PLoS ONE*, *9*(6). https://doi.org/10.1371/journal.pone.0099306

Bateman, A., Martin, M. J., O'Donovan, C., Magrane, M., Apweiler, R., Alpi, E., … Zhang, J. (2015). UniProt: A hub for protein information. *Nucleic Acids Research*, *43*(D1), D204–D212. https://doi.org/10.1093/nar/gku989

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., … Bourne, P. E. (2000). The protein data bank. *Nucleic Acids Research*, *28*(1), 235–242. https://doi.org/10.1093/nar/28.1.235

Bishop, J. M. (1991). Molecular themes in oncogenesis. *Cell*, *64*(2), 235–248. https://doi.org/10.1016/0092-8674(91)90636-D

Børresen-Dale, A. L. (2003). TP53 and breast cancer. *Human Mutation*, *21*(3), 292–300. https://doi.org/10.1002/humu.10174

Broderick, D. K., Di, C., Parrett, T. J., Samuels, Y. R., Cummins, J. M., McLendon, R. E., … Yan, H. (2004). Mutations of PIK3CA in anaplastic oligodendrogliomas, high-grade astrocytomas, and medulloblastomas. *Cancer Research*, *64*(15), 5048–5050.

https://doi.org/10.1158/0008-5472.CAN-04-1170

Campbell, I. G., Russell, S. E., Choong, D. Y. H., Montgomery, K. G., Ciavarella, M. L., Hooi, C. S. F., … Phillips, W. A. (2004). Mutation of the ***PIK3CA*** Gene in Ovarian and Breast Cancer. *Cancer Research*, *64*(21), 7678–7681. https://doi.org/10.1158/0008-5472.CAN-04-2933

Campbell, P. J., Martinocorena, I., & Campbell, P. J. (2015). Somatic mutation in cancer and normal cells. *Science*, *349*(6255), 1483–1489. https://doi.org/10.1126/science.aab4082

Cantley, L. C. (2002). The phosphoinositide 3-kinase pathway. *Science*, *296*(5573), 1655–1657. https://doi.org/10.1126/science.296.5573.1655

Cantley, L. C., & Neel, B. G. (1999). New insights into tumor suppression: PTEN suppresses tumor formation by restraining the phosphoinositide 3-kinase/AKT pathway. *Proceedings of the National Academy of Sciences*, *96*(8), 4240–4245. https://doi.org/10.1073/pnas.96.8.4240

Cantrell, D. a. (2001). Phosphoinositide 3-kinase signalling pathways. *Journal of Cell Science*, *114*(Pt 8), 1439–1445.

Carson, J. D., Van Aller, G., Lehr, R., Sinnamon, R. H., Kirkpatrick, R. B., Auger, K. R., … Luo, L. (2008). Effects of oncogenic p110α subunit mutations on the lipid kinase activity of phosphoinositide 3-kinase. *Biochemical Journal*, *409*(2), 519–524. https://doi.org/10.1042/BJ20070681

Carver, B. S., Chapinski, C., Wongvipat, J., Hieronymus, H., Chen, Y., Chandarlapaty, S., … Sawyers, C. L. (2011). Reciprocal Feedback Regulation of PI3K and Androgen Receptor Signaling in PTEN-Deficient Prostate Cancer. *Cancer Cell*, *19*(5), 575–586. https://doi.org/10.1016/j.ccr.2011.04.008

Chen, I.-C., Hsiao, L.-P., Huang, I.-W., Yu, H.-C., Yeh, L.-C., Lin, C.-H., … Lu, Y.-S. (2017). Phosphatidylinositol-3 Kinase Inhibitors, Buparlisib and Alpelisib, Sensitize Estrogen Receptor-positive Breast Cancer Cells to Tamoxifen. *Scientific Reports*, *7*(1), 9842. https://doi.org/10.1038/s41598-017-10555-z

Cheng, J. Q., Lindsley, C. W., Cheng, G. Z., Yang, H., & Nicosia, S. V. (2005). The Akt/PKB pathway: Molecular target for cancer drug discovery. *Oncogene*, *24*(50), 7482–7492. https://doi.org/10.1038/sj.onc.1209088

Cheung, L. W. T., Yu, S., Zhang, D., Li, J., Ng, P. K. S., Panupinthu, N., … Mills, G. B.

(2014). Naturally occurring neomorphic PIK3R1 mutations activate the MAPK pathway, dictating therapeutic response to MAPK pathway inhibitors. *Cancer Cell*, *26*(4), 479–494. https://doi.org/10.1016/j.ccell.2014.08.017

Croce, C. M. (1995). Oncogenes and cancer. *Science (New York, N.Y.)*, *267*(5203), 1408–1409. https://doi.org/10.1126/science.7878455

Cuevas, B. D., Lu, Y., Mao, M., Zhang, J., LaPushin, R., Siminovitch, K., & Mills, G. B. (2001). Tyrosine Phosphorylation of p85 Relieves Its Inhibitory Activity on Phosphatidylinositol 3-Kinase. *Journal of Biological Chemistry*, *276*(29), 27455–27461. https://doi.org/10.1074/jbc.M100556200

Cully, M., You, H., Levine, A. J., & Mak, T. W. (2006). Beyond PTEN mutations: The PI3K pathway as an integrator of multiple inputs during tumorigenesis. *Nature Reviews Cancer*, *6*(3), 184–192. https://doi.org/10.1038/nrc1819

Dixit, A., Yi, L., Gowthaman, R., Torkamani, A., Schork, N. J., & Verkhivker, G. M. (2009). Sequence and structure signatures of cancer mutation hotspots in protein kinases. *PLoS ONE*, *4*(10). https://doi.org/10.1371/journal.pone.0007485

Engelman, J. A., Chen, L., Tan, X., Crosby, K., Guimaraes, A. R., Upadhyay, R., … Wong, K. K. (2008). Effective use of PI3K and MEK inhibitors to treat mutant Kras G12D and PIK3CA H1047R murine lung cancers. *Nature Medicine*, *14*(12), 1351–1356. https://doi.org/10.1038/nm.1890

Forbes, S. A., Beare, D., Bindal, N., Bamford, S., Ward, S., Cole, C. G., … Campbell, P. J. (2016). COSMIC: High-resolution cancer genetics using the catalogue of somatic mutations in cancer. *Current Protocols in Human Genetics*, *2016*(October), 10.11.1-10.11.37. https://doi.org/10.1002/cphg.21

Fruman, D. a. (1998). Phosphoinositide Kinases. *Annu. Rev. Biochem.*, *67*, 481–507. https://doi.org/10.1146/annurev.biochem.67.1.481

Gabelli, S. B., Mandelker, D., Schmidt-Kittler, O., Vogelstein, B., & Amzel, L. M. (2010). Somatic mutations in PI3Kα: Structural basis for enzyme activation and drug design. *Biochimica et Biophysica Acta - Proteins and Proteomics*, *1804*(3), 533–540. https://doi.org/10.1016/j.bbapap.2009.11.020

Gao, J., Chang, M. T., Johnsen, H. C., Gao, S. P., Sylvester, B. E., Sumer, S. O., … Sander, C. (2017). 3D clusters of somatic mutations in cancer reveal numerous rare mutations as functional targets. *Genome Medicine*, *9*(1), 1–13.

https://doi.org/10.1186/s13073-016-0393-x

Greenman, C., Stephens, P., Smith, R., Dalgliesh, G. L., Hunter, C., Bignell, G., … Stratton, M. R. (2007). Patterns of somatic mutation in human cancer genomes. *Nature*, *446*(7132), 153–158. https://doi.org/10.1038/nature05610

Guo, X. E., Ngo, B., Modrek, A. S., & Lee, W.-H. (2014). Targeting tumor suppressor networks for cancer therapeutics. *Current Drug Targets*, *15*(1), 2–16. https://doi.org/10.1016/j.biotechadv.2011.08.021.Secreted

Hay, N. (2005). The Akt-mTOR tango and its relevance to cancer. *Cancer Cell*, *8*(3), 179–183. https://doi.org/10.1016/j.ccr.2005.08.008

Huang, C. H., Mandelker, D., Schmidt-Kittler, O., Samuels, Y., Velculescu, V. E., Kinzler, K. W., … Amzel, L. M. (2007). The structure of a human p110α/p85α complex elucidates the effects of oncogenic PI3Kα mutations. *Science*, *318*(5857), 1744–1748. https://doi.org/10.1126/science.1150799

Ikenoue, T., Kanai, F., Hikiba, Y., Obata, T., Tanaka, Y., Imamura, J., … Omata, M. (2005). Functional analysis of PIK3CA gene mutations in human colorectal cancer. *Cancer Research*, *65*(11), 4562–4567. https://doi.org/10.1158/0008-5472.CAN-04-4114

Izarzugaza, J. M. G., Redfern, O. C., Orengo, C. A., & Valencia, A. (2009). Cancer-associated mutations are preferentially distributed in protein kinase functional sites. *Proteins: Structure, Function and Bioinformatics*, *77*(4), 892–903. https://doi.org/10.1002/prot.22512

Kang, S., Bader, A. G., & Vogt, P. K. (2005). Phosphatidylinositol 3-kinase mutations identified in human cancer are oncogenic. *Proceedings of the National Academy of Sciences*, *102*(3), 802–807. https://doi.org/10.1073/pnas.0408864102

Karakas, B., Bachman, K. E., & Park, B. H. (2006). Mutation of the PIK3CA oncogene in human cancers. *British Journal of Cancer*, *94*(4), 455–459. https://doi.org/10.1038/sj.bjc.6602970

Katso, R., Okkenhaug, K., Ahmadi, K., Timms, J., & Waterfield, M. D. (2001). 3-K INASES : Implications for Development , Class I PI3Ks. *Annu. Rev. Cell Dev. Biol.*, *17*, 615–675. https://doi.org/10.1146/annurev.cellbio.17.1.615

Lee, J. O., Yang, H., Georgescu, M. M., Cristofano, A. Di, Maehama, T., Shi, Y., … Pavletich, N. P. (1999). Crystal structure of the PTEN tumor suppressor: Implications

for its phosphoinositide phosphatase activity and membrane association. *Cell*, *99*(3), 323–334. https://doi.org/10.1016/S0092-8674(00)81663-3

Levine, A. J. (1997). P53, the Cellular Gatekeeper for Growth and Division. *Cell*, *88*(3), 323–331. https://doi.org/10.1016/S0092-8674(00)81871-1

Liu, P., Cheng, H., Roberts, T. M., & Zhao, J. J. (2009). Targeting the phosphoinositide 3-kinase (PI3K) pathway in cancer. *Nat. Rev. Drug. Discov.*, *8*(8), 627–644. https://doi.org/10.1038/nrd2926.Targeting

LoRusso, P. M. (2016). Inhibition of the PI3K/AKT/mTOR pathway in solid tumors. *Journal of Clinical Oncology*, *34*(31), 3803–3815. https://doi.org/10.1200/JCO.2014.59.0018

Maehama, T., & Dixon, J. E. (1999). PTEN: A tumour suppressor that functions as a phospholipid phosphatase. *Trends in Cell Biology*, *9*(4), 125–128. https://doi.org/10.1016/S0962-8924(99)01519-6

Malanga, D., Scrima, M., De Marco, C., Fabiani, F., De Rosa, N., de Gisi, S., … Viglietto, G. (2008). Activating E17K mutation in the gene encoding the protein kinase AKT in a subset of squamous cell carcinoma of the lung. *Cell Cycle*, *7*(5), 665–669. https://doi.org/10.4161/cc.7.5.5485

Maley, C. C., Galipeau, P. C., Li, X., Sanchez, C. A., Paulson, T. G., & Reid, B. J. (2004). Selectively advantageous mutations and hitchhikers in neoplasms: p16 lesions are selected in Barrett's esophagus. *Cancer Research*, *64*(10), 3414–3427. https://doi.org/10.1158/0008-5472.CAN-03-3249

Mandelker, D., Gabelli, S. B., Schmidt-Kittler, O., Zhu, J., Cheong, I., Huang, C.-H., … Amzel, L. M. (2009). A frequent kinase domain mutation that changes the interaction between PI3Kalpha and the membrane. *Proceedings of the National Academy of Sciences of the United States of America*, *106*(40), 16996–17001. https://doi.org/10.1073/pnas.0908444106

Miled, N., Yan, Y., Hon, W. C., Perisic, O., Zvelebil, M., Inbar, Y., … Williams, R. L. (2007). Mechanism of two classes of cancer mutations in the phosphoinositide 3-kinase catalytic subunit. *Science*, *317*(5835), 239–242. https://doi.org/10.1126/science.1135394

mutation | Learn Science at Scitable. (n.d.). Retrieved January 30, 2018, from https://www.nature.com/scitable/definition/mutation-8

Myers, A. P., & Cantley, L. C. (2010). Targeting a common collaborator in cancer development. *Science Translational Medicine*, *2*(48). https://doi.org/10.1126/scitranslmed.3001251

National Breast Cancer Foundation. (n.d.). Retrieved January 30, 2018, from http://www.nationalbreastcancer.org/

Parsons, R. (2004). Human cancer, PTEN and the PI-3 kinase pathway. *Seminars in Cell and Developmental Biology*, *15*(2), 171–176. https://doi.org/10.1016/j.semcdb.2003.12.021

Pleasance, E. D., Cheetham, R. K., Stephens, P. J., McBride, D. J., Humphray, S. J., Greenman, C. D., … Stratton, M. R. (2010). A comprehensive catalogue of somatic mutations from a human cancer genome. *Nature*, *463*(7278), 191–196. https://doi.org/10.1038/nature08658

Pópulo, H., Lopes, J. M., & Soares, P. (2012). The mTOR signalling pathway in human cancer. *International Journal of Molecular Sciences*, *13*(2), 1886–1918. https://doi.org/10.3390/ijms13021886

Porta, C., Paglino, C., & Mosca, A. (2014). Targeting PI3K/Akt/mTOR Signaling in Cancer. *Frontiers in Oncology*, *4*(April), 1–11. https://doi.org/10.3389/fonc.2014.00064

Ruggero, D., & Sonenberg, N. (2005). The Akt of translational control. *Oncogene*, *24*(50), 7426–7434. https://doi.org/10.1038/sj.onc.1209098

Saal, L. H., Holm, K., Maurer, M., Merneo, L., Su, T., Wang, X., … Parsons, R. (2005). PIK3CA mutations correlate with hormonoreceptors, node metastasis and ERBB2 and are mutually exclusive with PTEN loss in human breast carcinoma. *Cancer Res*, *65*(7), 2554–2560. https://doi.org/10.1158/0008-5472-CAN-04-3913

Samuels, Y., Wang, Z., Bardelli, A., Silliman, N., Ptak, J., Szabo, S., … Velculescu, V. E. (2004). High Frequency of Mutations of the PIK3CA Gene in Human Cancers. *Science*, *304*(5670), 554. https://doi.org/10.1126/science.1096502

Slomovitz, B. M., & Coleman, R. L. (2012). The PI3K/AKT/mTOR pathway as a therapeutic target in endometrial cancer. *Clinical Cancer Research*, *18*(21), 5856–5864. https://doi.org/10.1158/1078-0432.CCR-12-0662

Stratton, M., Campbell, P., & Futreal, P. (2009). The cancer genome. *Nature*, *458*(7239), 719–724. https://doi.org/10.1038/nature07943.The

Tamborero, D., Gonzalez-Perez, A., & Lopez-Bigas, N. (2013). OncodriveCLUST: Exploiting the positional clustering of somatic mutations to identify cancer genes. *Bioinformatics*, *29*(18), 2238–2244. https://doi.org/10.1093/bioinformatics/btt395

Thorpe, L. M., Yuzugullu, H., & Zhao, J. J. (2015). PI3K in cancer: Divergent roles of isoforms, modes of activation and therapeutic targeting. *Nature Reviews Cancer*, *15*(1), 7–24. https://doi.org/10.1038/nrc3860

Vanhaesebroeck, B., & Waterfield, M. (1999). Signaling by distinct classes of phosphoinostide 3- kinases. *Experimental Cell Research*, *253*, 239–254.

Velankar, S., Dana, J. M., Jacobsen, J., Van Ginkel, G., Gane, P. J., Luo, J., … Kleywegt, G. J. (2013). SIFTS: Structure Integration with Function, Taxonomy and Sequences resource. *Nucleic Acids Research*, *41*(D1), 483–489. https://doi.org/10.1093/nar/gks1258

Vivanco, I., & Sawyers, C. L. (2002). The phosphatidylinositol 3-Kinase–AKT pathway in human cancer. *Nature Reviews Cancer*, *2*(7), 489–501. https://doi.org/10.1038/nrc839

Vogelstein, B., & Kinzler, K. W. (2004). Cancer genes and the pathways they control. *Nature Medicine*, *10*(8), 789–799. https://doi.org/10.1038/nm1087

Wan, X., Harkavy, B., Shen, N., Grohar, P., & Helman, L. J. (2007). Rapamycin induces feedback activation of Akt signaling through an IGF-1R-dependent mechanism. *Oncogene*, *26*(13), 1932–1940. https://doi.org/10.1038/sj.onc.1209990

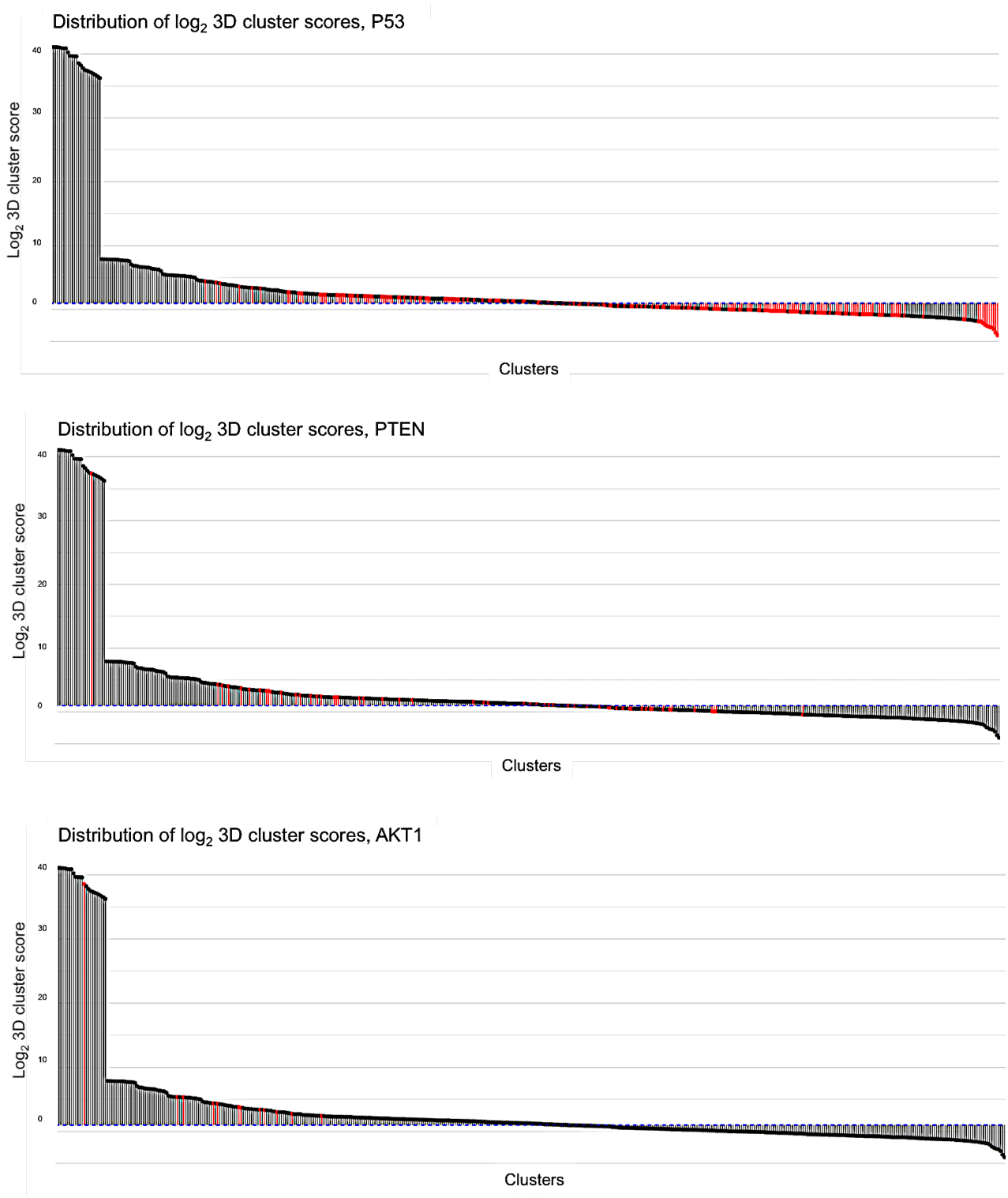
Wullschleger, S., Loewith, R., & Hall, M. N. (2006). TOR signaling in growth and metabolism. *Cell*, *124*(3), 471–484. https://doi.org/10.1016/j.cell.2006.01.016

Wymann, M. P., Zvelebil, M., & Laffargue, M. (2003). Phosphoinositide 3-kinase signalling - Which way to target? *Trends in Pharmacological Sciences*, *24*(7), 366–376. https://doi.org/10.1016/S0165-6147(03)00163-9

Ye, J., Pavlicek, A., Lunney, E. A., Rejto, P. A., & Teng, C. H. (2010). Statistical method on nonrandom clustering with application to somatic mutations in cancer. *BMC Bioinformatics*, *11*(2002). https://doi.org/10.1186/1471-2105-11-11

Yu, J., Wjasow, C., & Backer, J. M. (1998). Regulation of the p85/p110 ⎯ Phosphatidylinositol 3 □ -Kinase, *273*(46), 30199–30203.

Annex I



Annex 1. Distribution of log_2 3D cluster scores of the most significant 3D protein clusters in BRCA-mutated breast cancer dataset

Annex II

```
## 1- Breast_MAF.R
######################################################################
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Set directory
## 3. Load data - BRCA-mutated Breast cancer
## 4. Filter data
## 5. Save output
##
######################################################################
# Load Libraries
######################################################################
source("http://bioconductor.org/biocLite.R")
biocLite("maftools")
biocLite("Rsamtools")

# Load required libraries but suppressing the Package Startup Messages (this is in case you
want to embed this inside
# an automated script) to avoid flooding the terminal with non interesting messages.
suppressPackageStartupMessages(library(Rsamtools))
suppressPackageStartupMessages(library(maftools))
suppressPackageStartupMessages(library(dplyr))

# Load libraries
library(Rsamtools)
library(maftools)
library(dplyr)
library(data.table)

######################################################################
# Set directory
######################################################################

setwd("") #Set directory

######################################################################
# Load Data
######################################################################
###                          GDC                                   ###
###                  BRCA-mutated Breast Cancer Data               ###
######################################################################

start.time <- Sys.time()

### READ maf file
# INPUT: MAF file BRCA-mutated breast cancer data from: https://portal.gdc.cancer.gov/
maf_data <- maftools::read.maf(maf="./MAF/TCGA.BRCA.somatic.maf", removeSilent = F, useAll
= F)

# Save maf file as data frame
breast_data = maf_data@data

### FILTER and SORT data. Synonym and Non Synonym
#Sort data by Variant_Classification
breast_data = breast_data[with(breast_data, order(Variant_Classification))]
#Filter data by synonymous or non synonymous variation
synonym <- filter(breast_data, Variant_Classification == "Silent")
write.table(synonym, "./GDC-maf/BreastData_Synonym.csv", sep = "\t", row.names = FALSE) #
used in 2-Somatic_TSV.R
non_synonym <- filter(breast_data, Variant_Classification == "Missense_Mutation",
Variant_Type == 'SNP')
write.table(non_synonym, "./GDC-maf/BreastData_NonSynonym.csv", sep = "\t", row.names =
FALSE)
```

```
### Ensembl IDs (ENSP = idemtifiers for Proteins) list - ENSP  of the genes with Non
Synonymous mutations
# Data frame gene symbol, Ensembl ID, DNA and protein mutation position of non synonymous
variations
ns_df <- select(non_synonym, Hugo_Symbol, ENSP)
#DNA Change column
df <- select (non_synonym, Chromosome, Start_Position, Tumor_Seq_Allele1,
Tumor_Seq_Allele2)
ns_df$DNA_Change <- paste0(df$Chromosome, ":g.", df$Start_Position, df$Tumor_Seq_Allele1,
">", df$Tumor_Seq_Allele2)
#Protein Change columns
library(stringr)
split <- as.data.frame(str_match(non_synonym$HGVSp, "^(p).(.{3})(.*)(.{3})")[,-1])
# Add the two columns to ns_df
ns_df$HGVSp <- non_synonym$HGVSp_Short
ns_df$aa <- split$V2
ns_df$aa_mut <- split$V4
ns_df$prot_position <- split$V3

# Filter data frame, save unique values
ns_df_u <- ns_df[!duplicated(ns_df), ]
write.table(ns_df_u, "./NS_aa_position.csv", sep = "\t", row.names = FALSE)

# List of Ensembl IDs
ensembl_id <- unique(data.frame("ENSP_ID" = ns_df_u$ENSP))

# Save data frame and list with Ensembl IDs
write.csv(ensembl_id, "./EnsemblP_ID.csv", row.names = FALSE)  # used in 4-ENSP_PDB_NS.py

######################################################################
# Filter Data
######################################################################
#Mutation frequency
#Filter data by Non Synonymous variation
mutation_freq_NS <- fread('./Data/GDC/Breast/TSV/Mutation_freq/Breast_freq_NS.csv')

split_freq <- as.data.frame(str_match(mutation_freq_NS$Affected_cases_Cohort,
"^(.*)/(.*),(.*%)")[,-1])
split_freq2 <- as.data.frame(str_match(mutation_freq_NS$Affected_cases_GDC,
"^(.*)/(.*)")[,-1])
split_freq2$V2 <- gsub(',', '', split_freq2$V2)
mutation_freq_NS$Mutation_count <- split_freq$V1
mutation_freq_NS$Frequency_BRCA <- split_freq$V3
mutation_freq_NS$Total_cases_Cohort <- split_freq$V2
mutation_freq_NS$Frequency_GDC <- (as.integer(mutation_freq_NS$Mutation_count)/(as.integer(
split_freq2$V2))*100)
mutation_freq_NS$Frequency_GDC <- round(mutation_freq_NS$Frequency_GDC, 4)
mutation_freq_NS$Frequency_GDC <- paste(mutation_freq_NS$Frequency_GDC,'%')
mutation_freq_NS$Total_cases_GDC <- split_freq2$V2
mutation_freq_NS <- select(mutation_freq_NS, DNA_Change, Consequences, Mutation_count,
Frequency_BRCA, Total_cases_Cohort,
                           Frequency_GDC, Total_cases_GDC)

# MERGE
breast_NS <- merge(ns_df_u, mutation_freq_NS, by= c("DNA_Change"))

######################################################################
# Save Data
######################################################################
breast_NS <- select(breast_NS, Hugo_Symbol, ENSP, DNA_Change, HGVSp, aa, aa_mut,
prot_position, Consequences, Mutation_count,
                    Frequency_BRCA, Total_cases_Cohort, Frequency_GDC, Total_cases_GDC)
write.csv(breast_NS, "./Breast-NS_freq.csv", row.names = FALSE) # used in 4-ENSP_PDB_NS.py

end.time <- Sys.time()
time.taken <- end.time - start.time
print('Done!')
print(paste('Time:', time.taken, 'seconds'))
```

Annex III

```
## 2- Somatic_TSV.R
######################################################################
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Set directory
## 3. Load data - Synonymous mutations - background
## 4. Filter data
## 5. Load data - Non-synonymous mutations - BRCA-mutated Breast cancer
## 6. Merge data
## 7. Save output
##
######################################################################
# Load Libraries
######################################################################
suppressPackageStartupMessages(library(dplyr))
library(dplyr)
library(data.table)

######################################################################
#
# Set directory
#
######################################################################
setwd("")
######################################################################
# Load Data
######################################################################
###                          COSMIC                                ###
###           (Background - Synonymous mutations)                  ###
###             CosmicGenomeScreensMutantExport                    ###
######################################################################
start.time <- Sys.time()

#Big file --> 1.2GB
background <- fread('./COSMIC/CosmicGenomeScreensMutantExport.GRCh38.tsv') # (4452018
somatic mutations)

######################################################################
# Filter Data
######################################################################
# Data frame gene symbol, Ensembl ID, Patient ID, DNA and protein mutation position of
synonymous variations
background_S <- background[background$`Mutation Description` == 'Substitution - coding
silent']
# (1031615 synonym mutations)
background_S <- select(background_S, `Gene name`, `Accession Number`, `Sample name`,
`Mutation CDS`, `Mutation AA`,
                       `Mutation genome position`)
colnames(background_S) <- c('Gene_name', 'ENST', 'Sample_ID', 'Mutation_CDS', 'HGVSp',
'Mutation_genome_pos')

# Make variables from synonymous and non-synonymous files comparables
library(stringr)
split_gene_name <- as.data.frame(str_match(background_S$Gene_name, "^(.*)(_.*)")[,-1])
background_S$Hugo_Symbol <- as.character(split_gene_name$V1)
background_S$Hugo_Symbol[is.na(background_S$Hugo_Symbol)] <-
```

```
as.character(background_S$Gene_name[is.na

(background_S$Hugo_Symbol)])
split_chr_pos <- as.data.frame(str_match(background_S$Mutation_genome_pos, "^(.*):(.*)-
(.*)")[,-1])
split_mut <- as.data.frame(str_match(background_S$Mutation_CDS, "^(.*)(.>.)")[,-1])
background_S$DNA_Change <- paste('chr',split_chr_pos$V1,':g.',
split_chr_pos$V2,split_mut$V2, sep = "")

split_id <- as.data.frame(str_match(background_S$Sample_ID, "^(TCGA-.{2}-.{4})-(.{2})")[,-
1])
background_S$Patient_ID <- as.character(split_id$V1)
background_S$Patient_ID[is.na(background_S$Patient_ID)] <-
as.character(background_S$Sample_ID[is.na(background_S$Patient_ID)])
background_S <- select(background_S, 'Hugo_Symbol', 'ENST', 'DNA_Change', 'HGVSp',
'Patient_ID')

background_S_cases <- as.integer(length(unique(background_S$Patient_ID))) # (16328 cases)

######################################################################
# Load Data
######################################################################
#
###                            GDC                                 ###
###          Breast Data - Synonym mutations                       ###
#
######################################################################

Breast_synonym <- fread('./GDC-maf/BreastData_Synonym.csv') # From 1.A-Breast_MAF.R script

# Data frame gene symbol, Ensembl ID, DNA and protein mutation position of non synonymous
variations
s_df <- select(Breast_synonym, Hugo_Symbol, Transcript_ID)
colnames(s_df) <- c('Hugo_Symbol', 'ENST')
#DNA Change column
df <- select (Breast_synonym, Chromosome, Start_Position, Tumor_Seq_Allele1,
Tumor_Seq_Allele2)
s_df$DNA_Change <- paste0(df$Chromosome, ":g.", df$Start_Position, df$Tumor_Seq_Allele1,
">", df$Tumor_Seq_Allele2)
s_df$HGVSp <- Breast_synonym$HGVSp_Short
split_id_GDC <- as.data.frame(str_match(Breast_synonym$Matched_Norm_Sample_Barcode,
"^(TCGA-.{2}-.{4})-(.*)")[,-1])
s_df$Patient_ID <- as.character(split_id_GDC$V1)

######################################################################
# Merge Data
######################################################################
# Merge both datasets (TCGA BRCA synonym mutations that are not in COSMIC datset) -->
(#1044803)
not_common <- anti_join(background_S, s_df, by = c("DNA_Change", "Patient_ID")) ##1022412
Synonym <- merge(not_common, s_df, all = TRUE) #1044803

#library(plyr)#If you need functions from both plyr and dplyr, please load plyr first, then
dplyr.
Freq <- Synonym %>% plyr::count(c('DNA_Change', 'HGVSp'))

####### Add to S_freq hugo_symbol and ENST variables
S_freq <- merge (Freq, Synonym, by=c('DNA_Change', 'HGVSp'))
background_S_cases <- as.integer(length(unique(S_freq$Patient_ID))) # (16460 cases)
S_freq$Mutation_count <- S_freq$freq
S_freq$Frequency <- ((S_freq$freq/(as.integer(background_S_cases)))*100)
S_freq$Frequency <- round(S_freq$Frequency, 4)
```

```
S_freq$Frequency <- paste(S_freq$Frequency,'%')
S_freq$Total_cases_COSMIC <- background_S_cases

####### Convert aa names from 1 letter to 3
library(stringr)
split_aa <- as.data.frame(str_match(S_freq$HGVSp, "^(p).(.)(.*)(.)")[,-1])
code = data.frame(aa_1 = c('A', 'R', 'N', 'D', 'B', 'C', 'E', 'Q', 'Z', 'G', 'H', 'I', 'L',
'K', 'M', 'F', 'P', 'S',
                           'T', 'W', 'V', 'Y'),
                  aa_3 = c('ALA', 'ARG', 'ASN', 'ASP', 'ASX', 'CYS', 'GLU', 'GLN', 'GLX',
'GLY', 'HIS', 'ILE', 'LEU',
                           'LYS', 'MET', 'PHE','PRO', 'SER', 'THR', 'TRP', 'VAL', 'TYR'))

split_aa$V5 <-as.vector(code[match(split_aa$V2, code$aa_1),2])
split_aa$V6 <-  as.vector(code[match(split_aa$V4, code$aa_1),2])

S_freq$aa <- split_aa$V5
S_freq$aa_mut <- split_aa$V6
S_freq$prot_position <- split_aa$V3

S_freq <- select(S_freq, 'Hugo_Symbol', 'ENST', 'DNA_Change', 'HGVSp', 'aa', 'aa_mut',
'prot_position',
                 'Mutation_count', 'Frequency', 'Total_cases_COSMIC')
background_S_u <- S_freq[!duplicated(S_freq[,c('DNA_Change', 'HGVSp')]),]
write.csv(background_S_u, './Breast-S_freq.csv', row.names = FALSE)

# List of Ensembl IDs - ENST
ensembl_id <- data.frame("ENST_ID" = unique(S_freq$ENST)) # (29356 ENST) # used in 3-
ENST_PDB_S.py

######################################################################
# Save Data
######################################################################
# Save data frame and list with Ensembl IDs
write.csv(ensembl_id, "./EnsemblT_ID.csv", row.names = FALSE)  # used in 3-ENST_PDB_S.py

end.time <- Sys.time()
time.taken <- end.time - start.time
print('Done!')
print(paste('Time:', time.taken, 'seconds'))
```

Annex IV

```
## 3- ENST_PDB_S.py
# ------------------------------------------------------------------
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Load data - Ensembl ID and mutation counts synonymous mutations
## 3. Filter data
## 4. Merge data
## 5. Save data
##
# ------------------------------------------------------------------
# Load Libraries
# ------------------------------------------------------------------

from bioservices import UniProt
import numpy as np
import pandas as pd
from time import time
```

```
ini_time = time()
# ------------------------------------------------------------------
# Load data
# ------------------------------------------------------------------
Synonymous_df = pd.read_csv('./EnsemblT_ID.csv', sep='\t') #From 2-Somatic_TSV.R

#   Convert ENST ID column to a list
ensembl_id = Synonymous_df['ENST_ID'].tolist()
# ------------------------------------------------------------------
# ENST in UniProt
# ------------------------------------------------------------------
u = UniProt(verbose=False)

#   ID mapping UniProt
uniprot_id = u.mapping(fr = 'ENSEMBL_TRS_ID', to ='ACC', query = ensembl_id)
#print (uniprot_id)

#   Select only UniProt IDs
uniprot_dic = {}
uniprot_list = []
ensembl_id_2 = []

for element in uniprot_id:
    uniprot_dic[element] = uniprot_id[element][0]
    id = uniprot_id[element][0]
    uniprot_list.append(id)
    ensembl_id_2.append(element)

uniprot_inv_dic = {v: k for k, v in uniprot_dic.items()}

#   Save only non duplicate UniProt IDs
uniprot_list = set(uniprot_list)

#   NOT founded IDs function
def returnNotMatches(a, b):
    return [x for x in a if x not in b]

#   UniProt IDs not founded
uniprot_id_notfound = returnNotMatches(ensembl_id, ensembl_id_2)

# ------------------------------------------------------------------
# ENST-UniProt df
# ------------------------------------------------------------------
#   Save ENS-UniProt IDs in a df
ens_up_df = pd.DataFrame.from_dict(uniprot_dic, orient='index').reset_index()
ens_up_df.rename(columns={'index': 'ENST', 0 :'SP_PRIMARY'}, inplace=True)

# ------------------------------------------------------------------
# Merge Data
# ------------------------------------------------------------------
# Merge S_aa_position file (COSMIC y BRCA-GDC) with ENS-UniProt ID (COSMIC)
# by ENST, add UniProt column
ens_S_df = pd.read_csv('./Breast-S_freq.csv', sep=',', dtype = np.str) # From 2-
Somatic_TSV.R
ens_up_S_df = pd.merge(ens_S_df, ens_up_df, on=['ENST', 'ENST'])

# ------------------------------------------------------------------
# Save Data
# ------------------------------------------------------------------
ens_up_S_df.to_csv('/home/people/carlsa/TFM/Breast/Breast-S_freq-UP.csv', sep='\t', index=
False)
print('Breast-S_freq-UP: ',len(ens_up_S_df)) # Used in 6-PDB_S.R

fin_time = time()
print('Donde!')
print('Time: ', round(fin_time - ini_time, 6), "seconds")
```

Annex V

```
## 4- ENSP_PDB_NS.py
# ------------------------------------------------------------------
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Load data - Ensembl ID and mutation counts non-synonymous mutations
## 3. Filter data
## 4. Merge data
## 5. Save data
##
# ------------------------------------------------------------------
# Load Libraries
# ------------------------------------------------------------------

from bioservices import UniProt
import numpy as np
import pandas as pd
from time import time

ini_time = time()
# ------------------------------------------------------------------
# Load Data
# ------------------------------------------------------------------
Non_synonymous_df = pd.read_csv('./EnsemblP_ID.csv', sep='\t') #From 1-Breast_MAF.R

#   Convert ENSP ID column to a list
ensembl_id = Non_synonymous_df['ENSP_ID'].tolist()

# ------------------------------------------------------------------
# ENSP in UniProt
# ------------------------------------------------------------------
u = UniProt(verbose=False)

#   ID mapping UniProt
uniprot_id = u.mapping(fr = 'ENSEMBL_PRO_ID', to ='ACC', query = ensembl_id)

#   Select only UniProt IDs
uniprot_dic = {}
uniprot_list = []
ensembl_id_2 = []

for element in uniprot_id:
    uniprot_dic[element] = uniprot_id[element][0]
    id = uniprot_id[element][0]
    uniprot_list.append(id)
    ensembl_id_2.append(element)

uniprot_inv_dic = {v: k for k, v in uniprot_dic.items()}

#   Save only non duplicate UniProt IDs
uniprot_list = set(uniprot_list)

#   NOT founded IDs function
def returnNotMatches(a, b):
    return [x for x in a if x not in b]

#   UniProt IDs not founded
uniprot_id_notfound = returnNotMatches(ensembl_id, ensembl_id_2)

# ------------------------------------------------------------------
# ENS-UniProt Df
# ------------------------------------------------------------------
#   Save ENS-UniProt IDs in a Df
ens_up_df = pd.DataFrame.from_dict(uniprot_dic, orient='index').reset_index()
ens_up_df.rename(columns={'index': 'ENSP', 0 :'SP_PRIMARY'}, inplace=True)
```

```
print ('ENS_UP')

# ------------------------------------------------------------------
# Merge Data
# ------------------------------------------------------------------
# Merge NS_aa_position file (Breast data) with ENS-UniProt ID (Breast data)
# by ENSP, add UniProt column

ens_NS_df = pd.read_csv('./Breast-NS_freq.csv', sep=',', dtype = np.str) # From 1-
Breast_MAF.R
print('Breast-NS_freq: ',len(ens_NS_df))

# ------------------------------------------------------------------
# Save Data
# ------------------------------------------------------------------
ens_up_NS_df = pd.merge(ens_NS_df, ens_up_df, on=['ENSP', 'ENSP'])
ens_up_NS_df.to_csv('./Breast-NS_freq-UP.csv', sep='\t', index= False)
print('Breast-NS_freq-UP: ',len(ens_up_NS_df)) #Used in 5-PDB_NS.R

fin_time = time()
print('Done!')
print('Time: ', round(fin_time - ini_time, 6), "seconds")
```

Annex VI

```
## 5- PDB_NS.R
######################################################################
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Load data - RCSB and SIFTS files
## 3. Map mutation to PDB structures
## 4. Save data
##
######################################################################
# Load Libraries
######################################################################
# Load required libraries but suppressing the Package Startup Messages (this is in case you
want to embed this inside
# an automated script) to avoid flooding the terminal with non interesting messages.
suppressPackageStartupMessages(library(dplyr))
library(dplyr)
######################################################################
# Set directory
######################################################################
setwd()

######################################################################
# Load Data
######################################################################
###                  RCSB and SIFTS files                        ###
######################################################################
start.time <- Sys.time()

## PDB df filter by Homo sapiens from RCSB (https://www.rcsb.org/pdb)
pdb_Hs <- read.csv(file="./PDB_Hs/PDB_Hs.csv", sep = ',')
pdb_Hs <- pdb_Hs[pdb_Hs$Taxonomy.ID == '9606',]
pdb_Hs <- select(pdb_Hs, PDB.ID, Chain.ID, Exp..Method, Resolution)
colnames(pdb_Hs) <- c("PDB", "CHAIN", "Exp_Method", "Resolution")
```

```
pdb_Hs$wt <- 'mutant'

## PDB df tag wild type in wt column from RCSB
pdb_wild_ID <- read.table(file="./Data/PDB_Hs/PDB_wildtypeID.txt", sep = ',')
colnames(pdb_wild_ID)[1] <- 'ID'
pdb_Hs$wt[(as.vector(pdb_Hs$PDB) %in% as.vector(pdb_wild_ID$ID))]='wt'
levels(pdb_Hs$PDB) <- tolower(levels(pdb_Hs$PDB))

## Read tsv file with first and last positions of the proteins in the PDB structures from
SIFTS
## https://www.ebi.ac.uk/pdbe/docs/sifts/quick.html
pdb_pos <- read.table(file="./Data/SIFTS/pdb_chain_uniprot.tsv", skip = 1, header = TRUE,
sep = '\t')

## MERGE Homo sapiens PDB info (method) and SIFTS file (positions)
new = pdb_Hs[match(as.vector(pdb_pos$PDB),as.vector(pdb_Hs$PDB)),]
pdb_data = cbind(pdb_pos,new)

## Match one mutation with one PDB file
mutation <- read.table(file="./Breast-NS_freq-UP.csv", header = TRUE, sep = '\t') #From 4-
ENSP_PDB_NS.py
print(paste('Breast-NS_freq-UP:', nrow(mutation)))

######################################################################
# Map mutations to PDB structures
######################################################################
pdb_pos_wt = pdb_data
listUP=unique(pdb_pos_wt$SP_PRIMARY)
mutation$best_pdb = "none"
mutation$Chain = "none"
mutation$Exp.Method = "none"
mutation$Resolution = "none"
mutation$PDB_beg = "none"
mutation$PDB_end = "none"
mutation$UP_beg = "none"
mutation$UP_end = "none"
mutation$PDB_mpos = "none"
mutation$PDB_aapos = "none"

x <- sapply(mutation, is.factor)
mutation[x] <- lapply(mutation[x], as.character)

x <- sapply(pdb_pos_wt, is.factor)
pdb_pos_wt[x] <- lapply(pdb_pos_wt[x], as.character)

for (i in (1:nrow(mutation))){
  print(i)
  #print (mutation$SP_PRIMARY[i])
  pdb_up = pdb_pos_wt[pdb_pos_wt$SP_PRIMARY == mutation$SP_PRIMARY[i],]
  #print (pdb_up)
  if (nrow(pdb_up)==0){
    next
  }
  else {
    filter=c()
    for (x in (1:nrow(pdb_up))){
      filter=c(filter,((mutation$prot_position[i] > pdb_up$SP_BEG[x]) &
(mutation$prot_position[i] < pdb_up$SP_END[x])))
    }
    pdb_up_pos = pdb_up[filter,]
  }
```

```
  if (nrow(pdb_up_pos)==0){
    next
  }
  else{
    ranges=pdb_up_pos$SP_END-pdb_up_pos$SP_BEG
    index=which(ranges==max(pdb_up_pos$SP_END-pdb_up_pos$SP_BEG))
    pdb_up_pos_range=pdb_up_pos[index,]
  }
  # Select PDB structure between more than one possibility:
  # Experimental method: X-ray
  # Highest resolution (lowest value)
  # Longer protein sequence (highest difference between begining and ending protein
sequence positions)
  if ((nrow(pdb_up_pos_range)>1) & ("X-RAY" %in% pdb_up_pos_range$Exp_Method)){
    indexExp=which(pdb_up_pos_range$Exp_Method=="X-RAY DIFFRACTION")
    pdb_up_pos_range_exp=pdb_up_pos_range[indexExp,]

    if (nrow(db_up_pos_range_exp)>1){
      res=pdb_up_pos_range_exp$Resolution
      indexRes=which(res==min(res))
      pdb_up_pos_range_exp_res=db_up_pos_range_exp[indexRes,]

      mutation$best_pdb[i] = pdb_up_pos_range_exp_res$PDB
      mutation$Chain[i] = pdb_up_pos_range_exp_res$CHAIN[1]
      mutation$Exp.Method[i] = pdb_up_pos_range_exp_res$Exp_Method
      mutation$Resolution[i] = pdb_up_pos_range_exp_res$Resolution
      mutation$PDB_beg[i] = pdb_up_pos_range_exp_res$RES_BEG[1]
      mutation$PDB_end[i] = pdb_up_pos_range_exp_res$RES_END[1]
      mutation$UP_beg[i] = pdb_up_pos_range_exp_res$SP_BEG[1]
      mutation$UP_end[i] = pdb_up_pos_range_exp_res$SP_END[1]
      mutation$PDB_mpos[i] = ((as.integer(mutation$PDB_beg[i])-
as.integer(mutation$UP_beg[i])) +
                              (as.integer(mutation$prot_position[i])))
      mutation$PDB_aapos[i] = paste(toupper(mutation$aa[i]),mutation$PDB_mpos[i], sep = "")
    }

    else{
      mutation$best_pdb[i] = pdb_up_pos_range_exp$PDB
      mutation$Chain[i] = pdb_up_pos_range_exp$CHAIN[1]
      mutation$Exp.Method[i] = pdb_up_pos_range_exp$Exp_Method
      mutation$Resolution[i] = pdb_up_pos_range_exp$Resolution
      mutation$PDB_beg[i] = pdb_up_pos_range_exp$RES_BEG[1]
      mutation$PDB_end[i] = pdb_up_pos_range_exp$RES_END[1]
      mutation$UP_beg[i] = pdb_up_pos_range_exp$SP_BEG[1]
      mutation$UP_end[i] = pdb_up_pos_range_exp$SP_END[1]
      mutation$PDB_mpos[i] = ((as.integer(mutation$PDB_beg[i])-
as.integer(mutation$UP_beg[i])) +
                              (as.integer(mutation$prot_position[i])))
      mutation$PDB_aapos[i] = paste(toupper(mutation$aa[i]),mutation$PDB_mpos[i], sep = "")
    }
  }
  else{
    pdb_up_pos_range=pdb_up_pos[(index[1]),]
    mutation$best_pdb[i] = pdb_up_pos_range$PDB[1]
    mutation$Chain[i] = pdb_up_pos_range$CHAIN[1]
    mutation$Exp.Method[i] = pdb_up_pos_range$Exp_Method[1]
    mutation$Resolution[i] = pdb_up_pos_range$Resolution[1]
    mutation$PDB_beg[i] = pdb_up_pos_range$RES_BEG[1]
    mutation$PDB_end[i] = pdb_up_pos_range$RES_END[1]
    mutation$UP_beg[i] = pdb_up_pos_range$SP_BEG[1]
    mutation$UP_end[i] = pdb_up_pos_range$SP_END[1]
    mutation$PDB_mpos[i] = ((as.integer(mutation$PDB_beg[i])-
```

```
as.integer(mutation$UP_beg[i])) +
                                (as.integer(mutation$prot_position[i])))
    mutation$PDB_aapos[i] = paste(toupper(mutation$aa[i]),mutation$PDB_mpos[i], sep = "")

  }

}

#####################################################################
# Save data
#####################################################################
# Save mutation data mapped with PDB information
write.csv(mutation, "./Breast-NS_freq-UP-PDB.csv", row.names = FALSE)
print(paste('Breast-NS_freq-UP-PDB:', nrow(mutation)))

# PDB data
PDB_list_NS <- select(mutation,best_pdb,Chain)
colnames(PDB_list_NS) <- c('ID', 'Chain')

# sort file
PDB_list_NS <- PDB_list_NS[order(PDB_list_NS$ID, PDB_list_NS$Chain),]
PDB_list_NS <- (unique(PDB_list_NS))
PDB_list_NS <-PDB_list_NS [! PDB_list_NS$ID %in% 'none',]

# Save PDB list to download PDB files (6-PDB-download.py)
PDB_id_NS <- data.frame("ID" = PDB_list_NS$ID)
PDB_id_NS  <- unique(PDB_id_NS)
write.csv(PDB_id_NS, "./PDB_ID_NS.csv", row.names = FALSE)
print(paste('PDB_ID_NS:', nrow(PDB_id_NS)))

# Save PDB-chain list to calculate 3D clusters (7-Euclidean_distance.py)
write.csv(PDB_list_NS, "./PDB_Chain_NS.csv", row.names = FALSE)
print(paste('PDB_Chain_NS.csv:', nrow(PDB_list_NS)))

end.time <- Sys.time()
time.taken <- end.time - start.time
print('Done!')
print(paste('Time:', time.taken,'seconds'))
```

Annex VII

```
## 7- PDB_download.py
# ------------------------------------------------------------------
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Load data - PDB IDs
## 3. Download data
## 5. Save data
##
# ------------------------------------------------------------------
# Load Libraries
# ------------------------------------------------------------------
import os
from Bio.PDB import *
from bioservices import UniProt
import numpy as np
import pandas as pd
from time import time
import os.path as path
```

```
# ------------------------------------------------------------
# Load Data
# ------------------------------------------------------------
PDB_ID = pd.read_csv('./PDB_ID_NS.csv') # From 5-PDB_NS.R
#   Convert ENSP ID column to a list
PDB_IDList = PDB_ID.ID.tolist()

# ------------------------------------------------------------
#
# Download ans Save PDB files (.cif format)
#
# ------------------------------------------------------------

ini_time_total = time()
not_download = list()
for i in range(0,  len(PDB_IDList)):
    name = PDB_IDList[i]
    print('------------------------------------------------------------')
    print(name)
    file = '/home/people/carlsa/TFM/Breast/PDB2/' + name + '.cif'
    if path.exists(file):
        print('Preload file:', name)
        print((i + 1), '/', len(PDB_IDList), )
        continue

    else:
        print('Loading file: ', name)
        ini_time = time()
        pdbl = PDBList()

pdbl.retrieve_pdb_file(pdb_code=name,file_format="mmCif",pdir="/home/people/carlsa/TFM/Brea
st/PDB2/")
        if path.exists(file):
            print('Loaded file: ', name)
        else:
            print('ERROR. Unloaded file: ', name)
            not_download.append(name)
        fin_time = time()
        print('Time: ', round(fin_time - ini_time, 6), "seg")
        print ((i+1), '/', len(PDB_IDList),)
print ('------------------------------------------------------------')
print ('NOT downloaded PDB files: ', not_download)
not_download_df = pd.DataFrame(not_download, columns=["PDB"])
# PDB files that cannot be downloaded
not_download_df.to_csv('./Error_PDB.csv', sep='\t', index= False)

fin_time_total = time()
print('Total time : ', round(fin_time_total - ini_time_total, 3), "seg")
```

Annex VIII

```
# 8- Euclidean_dist.py
#  Euclidean distance computation to generate 3D protein clusters
# we used atom coordinates from PDB.cif files
# ------------------------------------------------------------
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Load data - PDB files
## 3. Euclidean distance computation
## 5. Save data
##
# ------------------------------------------------------------
# Load Libraries
# ------------------------------------------------------------
import os
```

```
import pandas as pd
import numpy as np
from time import time
import os.path as path

# ----------------------------------------------------------------
# Load Data
# ----------------------------------------------------------------
PDB_df = pd.read_csv('./PDB_Chain_NS.csv',  sep=',') #From 5-PDB_NS.R

#   Convert ENSP ID column to a list
PDB_ID = PDB_df['ID'].tolist()

# ----------------------------------------------------------------
# Euclidean distance computationLoad Data
# ----------------------------------------------------------------
not_pdb = list()
for i in range(0, len(PDB_df)):
    name_pdb = PDB_ID[i]
    chain = str(PDB_df['Chain'][i])
    print('-----------------------------------------------------------------')
    print('Loading file: ', name_pdb)
    print(i, '/', (len(PDB_df)-1))
    PDB_file ='/home/people/carlsa/TFM/Breast/PDB/' + name_pdb +'.cif'
    name_pdb = name_pdb.upper()
    eucl_file = '/home/people/carlsa/TFM/Breast/3D_cluster/' + name_pdb + '_' + chain +
'.txt'
    if path.exists(eucl_file):
        print('Pre-calculated 3D cluster for file :', (name_pdb + '_' + chain))
        continue

    if not path.exists (PDB_file):
        print('ERROR. File ', name_pdb, ' NOT found')
        not_pdb.append(name_pdb)
        continue
    #Select the variables in PDB.cif file
    else :
        from Bio.PDB.MMCIFParser import MMCIFParser
        parser = MMCIFParser()
        from Bio.PDB.MMCIF2Dict import MMCIF2Dict
        try:
            mmcif_dict = MMCIF2Dict(PDB_file)
            # .cif files are indexed
            # Select coulmns of interest
            # Add each column to a variable

            name = mmcif_dict['_entry.id']
            print(name)

            # Strands of the protein in the PDB file
            strand = mmcif_dict['_entity_poly.pdbx_strand_id']
            print('strand: ', strand)

            #name of the protein
            entity_id = mmcif_dict['_entity_poly.entity_id']
            entity_id_2 = mmcif_dict['_atom_site.label_entity_id']
            strct_letter = mmcif_dict['_struct_asym.id']
            strct_number = mmcif_dict['_struct_asym.entity_id']
            loop_number = mmcif_dict['_atom_site.pdbx_PDB_model_num']
            group = mmcif_dict['_atom_site.group_PDB']
            # name of the atom of the amino-acid
            atom = mmcif_dict['_atom_site.label_atom_id']
            # symbol of the atom
            prog_num = mmcif_dict['_atom_site.id']
            alt_conformation = mmcif_dict['_atom_site.label_alt_id']
            # amino-acid
            resid = mmcif_dict['_atom_site.label_comp_id']
            #amino-acid residue number
            pos = mmcif_dict['_atom_site.label_seq_id']
            # amino-acid residue number in PDB file
            pos_pymol = mmcif_dict['_atom_site.auth_seq_id']
```

```
            # name of the aminacid
            string = mmcif_dict['_atom_site.label_asym_id']
            #Residue coordinates
            x_list = mmcif_dict['_atom_site.Cartn_x']
            y_list = mmcif_dict['_atom_site.Cartn_y']
            z_list = mmcif_dict['_atom_site.Cartn_z']

            if type(strand) == str:
                aux = list()
                aux.append(strand)
                strand = aux

            # Look for the correct strand in the PDB file:
            flag = 0
            chain_match = 0
            for j in range(0, len(entity_id)):
                if flag == 1:
                    break
                else:
                    strand_pdb = list(strand[j].split(','))
                    for k in range(0, len(strand_pdb)):
                        if strand_pdb[k] == chain:
                            flag = 1
                            chain_match = entity_id[j]
                            print('Match Chain - Entity id: ', chain, ', ', chain_match)
                            break

            # Crate a data frame with all the mutations and coordinates information
            df_coord = pd.DataFrame({"Group": group, "Alt_Conformation": alt_conformation,
"ATOM": atom, "Residue": resid, "Position": pos, "Position_PyMOL": pos_pymol, "String":
string, "Entity_id": entity_id_2, "Loop_number": loop_number, "coord_x": x_list, "coord_y":
y_list, "coord_z": z_list})


            string_u = strct_letter[strct_number.index(chain_match)]
            loop_number_2 = df_coord.Loop_number.unique()
            # Filter dataframe by CA (alfa carbons), chain, string,....
            df_coord = df_coord.loc[df_coord['Group'] == 'ATOM']
            df_coord = df_coord.loc[df_coord['ATOM'] == 'CA']
            df_coord = df_coord.loc[df_coord['Entity_id'] == chain_match]
            df_coord = df_coord.loc[df_coord['String'] == string_u]
            df_coord = df_coord.loc[df_coord['Loop_number'] == loop_number_2[0]]
            df_coord = df_coord.loc[df_coord['Alt_Conformation'].isin(['.', 'A'])]


            # EUCLIDEAN DISTANCE COMPUTATION
            # Create a data frame to save euclidean distance between two alfa carbons (CA)
            # Filter by residue index, no euclidean distance between same residue
            # Filter by euclidean distance value below 15 Amstrongs --> 3D clusters
            df_dist = pd.DataFrame(columns=('Chain', 'Entity_id', 'string_x', 'aa_x',
'pos_x', 'string_y', 'aa_y', 'pos_y',
                                            'euclidean_dist', 'pos_PyMOLx', 'pos_PyMOLy'))

            ini_time = time()
            for x in range(len(df_coord)):
                for y in range(len(df_coord)):
                    if (df_coord.index[x] != df_coord.index[y]):
                        x_cor_1 = float(df_coord.coord_x.iloc[x])
                        y_cor_1 = float(df_coord.coord_y.iloc[x])
                        z_cor_1 = float(df_coord.coord_z.iloc[x])
                        x_cor_2 = float(df_coord.coord_x.iloc[y])
                        y_cor_2 = float(df_coord.coord_y.iloc[y])
                        z_cor_2 = float(df_coord.coord_z.iloc[y])
                        x_diff = x_cor_1 - x_cor_2
                        y_diff = y_cor_1 - y_cor_2
                        z_diff = z_cor_1 - z_cor_2
                        euclidean_diff = (((x_diff ** 2) + (y_diff ** 2) + (z_diff ** 2))
** 0.5)
                        #print(euclidean_diff)
                        if euclidean_diff <= float(15):
                            df_dist.loc[len(df_dist)] = [chain, df_coord.Entity_id.iloc[x],
```

```
df_coord.String.iloc[x], df_coord.Residue.iloc[x], df_coord.Position.iloc[x],
df_coord.String.iloc[y], df_coord.Residue.iloc[y], df_coord.Position.iloc[y],
euclidean_diff,df_coord.Position_PyMOL.iloc[x], df_coord.Position_PyMOL.iloc[y]]

            fin_time = time()
            print('Done!')
            print('Time: ', round(fin_time - ini_time, 6), "seg")
# ----------------------------------------------------------------
# Save Data
# ----------------------------------------------------------------
            df_dist.to_csv('./3D_cluster/' + name + '_' + chain + '.txt', sep='\t',
index=False) #Used in 9-3Dcluster.R
        except ValueError:
            print('ERROR in file ', name_pdb)
            not_pdb.append(name_pdb)
            continue

print ('-----------------------------------------------------------------')
print ('NOT founded PDB files: ', not_pdb)
not_pdb_df = pd.DataFrame(not_pdb, columns= ["PDB"])
not_pdb_df.to_csv("./euclidistance_ERROR.csv", sep='\t',index=False)
```

Annex IX

```
## 9-3Dcluster.R
# Identify mutated residues in 3D protein clusters and asign NS ans S mutations counts
#######################################################################
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Load data
## 3. 3D protein cluster computation
## 4. Save data
##
#######################################################################
# Load Libraries
#######################################################################
suppressPackageStartupMessages(library(dplyr))
library(dplyr)
#######################################################################
# Set directory
#######################################################################
setwd("")

#######################################################################
# Load Data
#######################################################################
## PDB 3D cluster file
pdb_idlist <- read.csv(file="./PDB_Chain_NS.csv", sep = ',') # From 5-PDB_NS.R
mutation_df_NS <- fread(file = "./Breast-NS_freq-UP-PDB.csv", sep = ',') # From 5-PDB_NS.R
mutation_df_S <- fread(file = "./Breast-S_freq-UP-PDB.csv", sep = ',') # From 6-PDB_S.R

#######################################################################
#3D protein cluster computation
#######################################################################
# Identify mutated residues in 3D clusters and asign NS and S mutant counts
error_list = c()
for (i in (1:nrow(pdb_idlist))){
  i = 4
  id = as.vector(pdb_idlist$ID[i])
  chain = as.vector(pdb_idlist$Chain[i])
```

```
  ID = toupper(id)
  filename = paste(ID,chain, sep = "_")
  input = paste("./3D_cluster/",filename,".txt", sep = "")
  output_freq = paste("./3D_cluster_freq/", filename, "_freq", ".txt", sep = "")
  print ('------------------------------------------------------------')
  print(paste('Loading file:', filename))
  if (file.exists(output_freq)){
    print(paste('Pre-calculated 3D cluster for file:', filename))
    print(paste(i, '/', nrow(pdb_idlist)))
    next
  } else{
    print(paste(i, '/', nrow(pdb_idlist)))
    if (!file.exists(input)){
      error_list = c(error_list, filename)
      print(paste('File', filename, 'NOT found'))
      next
    } else {
      pdb_3D <- read.csv(file=input, sep = '\t')
      print(paste('Calculating 3D cluster for file:', filename))
      # Residue position in PDB
      res_pos_x = paste(pdb_3D$aa_x,pdb_3D$pos_x,"_",pdb_3D$Chain, sep = "")
      pdb_3D$res_pos_x <- res_pos_x
      res_pos_y = paste(pdb_3D$aa_y,pdb_3D$pos_y,"_",pdb_3D$Chain, sep = "")
      pdb_3D$res_pos_y <- res_pos_y

      # Residue position in PyMOL
      res_pos_PyMOLx = paste(pdb_3D$aa_x,pdb_3D$pos_PyMOLx,"_",pdb_3D$Chain, sep = "")
      pdb_3D$res_pos_PyMOLx <- res_pos_PyMOLx
      res_pos_PyMOLy = paste(pdb_3D$aa_y,pdb_3D$pos_PyMOLy,"_",pdb_3D$Chain, sep = "")
      pdb_3D$res_pos_PyMOLy <- res_pos_PyMOLy
    }
  }
  dataset = pdb_3D
  res = unique(dataset$res_pos_x)
  res_PyMOL = unique(dataset$res_pos_PyMOLx)
  cluster = list()
  cluster_PyMOL = list()
  for (i in (1:length(res))){
    # Cluster file, all residue names + position in PDB for each cluster
    res_clust = res[i]
    pdb_clust = dataset[dataset$res_pos_x == res_clust,]
    cluster = c(cluster, list(c(res_clust,pdb_clust$res_pos_y)))
    len_cluster <- sapply(cluster, length)
    max_cluster <- seq_len(max(len_cluster))
    matrix_cluster <- t(sapply(cluster, "[", i=max_cluster))
    output_cluster = paste("./3D_cluster_freq/", filename, "_cluster", ".txt", sep = "")
    write.csv(matrix_cluster, output_cluster, row.names = FALSE)

    # Cluster PyMOL file, all residue names + position in PyMOL for each cluster
    res_clust_PyMOL = res_PyMOL[i]
    pdb_clust_PyMOL = dataset[dataset$res_pos_PyMOLx == res_clust_PyMOL,]
    cluster_PyMOL = c(cluster_PyMOL,
list(c(res_clust_PyMOL,pdb_clust_PyMOL$res_pos_PyMOLy)))
    len_cluster_PyMOL <- sapply(cluster_PyMOL, length)
    max_cluster_PyMOL <- seq_len(max(len_cluster_PyMOL))
    matrix_cluster_PyMOL <- t(sapply(cluster_PyMOL, "[", i=max_cluster_PyMOL))
    output_cluster_2 = paste("./3D_cluster_freq/", filename, "_clusterPyMOL", ".txt", sep =
"")
    write.csv(matrix_cluster_PyMOL, output_cluster_2, row.names = FALSE)

    # Cluster lenght file
    summary(cluster)
```

```
    cluster_summary = summary(cluster)
    cluster_len = as.data.frame.matrix(cluster_summary)
    cluster_len = as.vector(cluster_len$Length)
    output_len = paste("./3D_cluster_freq/", filename, "_lenght", ".txt", sep = "")
    write.csv(cluster_len, output_len, row.names = FALSE)
  }

  # NS mutant counts
  mut_pdb_NS=mutation_df_NS[((mutation_df_NS$best_pdb==id) &
(mutation_df_NS$Chain==chain)),]
  freq_NS =list()
  print(paste('Calculating Non-synonymous mutation frequencies for file:', filename))
  for (i in (1:nrow(cluster_summary))){
    temp_res_NS=c()
    for (r in (cluster[[i]])){
      if (unlist(strsplit(r,"_"))[1] %in% as.vector(mut_pdb_NS$PDB_aapos)){
        for (y in (1:nrow(mut_pdb_NS))){
          if (unlist(strsplit(r,"_"))[1]==as.vector(mut_pdb_NS$PDB_aapos[y])){

temp_res_NS=c(temp_res_NS,as.integer(unlist(as.vector(mut_pdb_NS$Mutation_count[y])))[1])
          }
        }
      }
      else{
        temp_res_NS=c(temp_res_NS,0)
      }
    }
    freq_NS = c(freq_NS,list(temp_res_NS))
  }
  len_freq_NS <- sapply(freq_NS, length)
  max_freq_NS <- seq_len(max(len_freq_NS))
  matrix_freq_NS <- t(sapply(freq_NS, "[", i=max_freq_NS))
  output_freq_NS = paste("./3D_cluster_freq/", filename, "_freq_NS", ".txt", sep = "")
  write.csv(matrix_freq_NS, output_freq_NS, row.names = FALSE)
  summary(freq_NS)
  freq_NS_summary = summary(freq_NS)

  sum_list_NS=c()
  for (t in (1:nrow(cluster_summary))){
    sum_list_NS=c(sum_list_NS,sum(freq_NS[[t]]))
  }
  print ('Summary')
  print (summary(sum_list_NS))
  print (nrow(mut_pdb_NS))


  # S mutant counts
  mut_pdb_S=mutation_df_S[((mutation_df_S$best_pdb==id) & (mutation_df_S$Chain==chain)),]
  freq_S =list()
  print(paste('Calculating Synonymous mutation frequencies for file:', filename))
  for (i in (1:nrow(cluster_summary))){
    temp_res_S=c()
    for (r in (cluster[[i]])){
      if (unlist(strsplit(r,"_"))[1] %in% as.vector(mut_pdb_S$PDB_aapos)){
        for (y in (1:nrow(mut_pdb_S))){
          if (unlist(strsplit(r,"_"))[1]==as.vector(mut_pdb_S$PDB_aapos[y])){

temp_res_S=c(temp_res_S,as.integer(unlist(as.vector(mut_pdb_S$Mutation_count[y])))[1])
          }
        }
      }
      else{
```

```
        temp_res_S=c(temp_res_S,0)
      }
    }
    freq_S = c(freq_S,list(temp_res_S))
  }
  len_freq_S <- sapply(freq_S, length)
  max_freq_S <- seq_len(max(len_freq_S))
  matrix_freq_S <- t(sapply(freq_S, "[", i=max_freq_S))
  output_freq_S = paste("./3D_cluster_freq/", filename, "_freq_S", ".txt", sep = "")
  write.csv(matrix_freq_S, output_freq_S, row.names = FALSE)
  summary(freq_S)
  freq_S_summary = summary(freq_S)

  sum_list_S=c()
  for (t in (1:nrow(cluster_summary))){
    sum_list_S=c(sum_list_S,sum(freq_S[[t]]))
  }
  print ('Summary')
  print (summary(sum_list_S))
  print (nrow(mut_pdb_S))
  frequencies = data.frame(freq_NS = sum_list_NS, freq_S = sum_list_S)
  write.csv(frequencies, output_freq, row.names = FALSE)

}
print ('------------------------------------------------------------')
print (paste('Input files NOT found:', error_list))
write.csv(error_list, './Error_3Dcluster.csv', row.names = FALSE)

end.time <- Sys.time()
time.taken <- end.time - start.time
print('Done!')
print(paste('Time:', time.taken, 'seconds))
```

Annex X

```
## 10-FreqLen.R
#######################################################################
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
##
## 1. Load libraries
## 2. Load data - Frequency of NS and S and cluster lenght
## 3. Merge data
## 4. Save as .RData
##
#######################################################################
# Load Libraries
#######################################################################
suppressPackageStartupMessages(library(dplyr))
library(dplyr)
library(stringr)
#######################################################################
# Set directory
#######################################################################
setwd("")

#######################################################################
# Load Data
#######################################################################
```

```
###     Frequency of NS and S and cluster lenght file          ###
######################################################################
start.time <- Sys.time()

pdb_list <- read.csv('./PDB_Chain_NS.csv') # From 5-PDB_NS.R

######################################################################
# Merge Data
######################################################################
# Create error list
error_list = c()
freq_len <- data.frame()
for (i in (1:nrow(pdb_list))){
  id = as.vector(pdb_list$ID[i])
  chain = as.vector(pdb_list$Chain[i])
  ID = toupper(id)
  filename = paste(ID,chain, sep = "_")
  input = paste("./3D_cluster_freq_B/",filename,"_freq.txt", sep ="")
  input_2 = paste("./3D_cluster_freq_B/",filename,"_lenght.txt", sep ="")
  output = paste("./3D_cluster_freq_B/",filename,"_freqlen.txt", sep ="")
  print ('------------------------------------------------------------')
  print(paste('Loading file:', filename))
  if (file.exists(input) & file.exists(input_2)){
    freq_df <- read.csv(file=input)
    len_df <- read.csv(file = input_2, col.names = "length")
    freq_file <- data.frame(cluster = 1:nrow(freq_df))
    freq_file$PDB <- filename
    freq_file$mut_count_NS <- freq_df$freq_NS
    freq_file$mut_count_S <- freq_df$freq_S
    freq_file$length <- len_df$length
    write.csv(freq_file, output, row.names = FALSE)
  } else {
    error_list = c(error_list, filename)
    print(paste('File', filename, 'NOT found'))
    next
  }
  freq_len <- rbind(freq_len, freq_file)
}

######################################################################
# Save Data
######################################################################

write.csv(freq_len, "./FreqLen.csv", row.names = FALSE) # used in 11-score.R
print ('------------------------------------------------------------')
print (paste('Input files NOT found:', error_list))
write.csv(error_list, './TFM_FreqLen_ERROR.csv', row.names = FALSE)

end.time <- Sys.time()
time.taken <- end.time - start.time
print('Done!')
print(paste('Time:', time.taken, 'seconds'))
```

Annex XI

```
## 11-Score.R
######################################################################
## DTU - Department of Bio and Health Informatics
## 2017
## Carmen Saenz
```

```
##
## 1. Load libraries
## 2. Load data - Frequency of NS and S and cluster lenght
## 3. 3D cluster score computation
## 4. Save as .RData
##
######################################################################
# Load Libraries
######################################################################
suppressPackageStartupMessages(library(dplyr))
library(data.table)
library(stringr)

######################################################################
# Set directory
######################################################################
setwd("")
#############################################################################

######################################################################
# Load Data
######################################################################
###     Frequency of NS and S and cluster lenght file           ###
######################################################################
start.time <- Sys.time()

# READ files
mutcount = fread(file = "./FreqLen.csv", sep = ',') #(1125683 clusters) #From 10-FreqLen.R
NS_mutation_df = fread(file = "./Breast-NS_freq-UP-PDB.csv", sep = ",") #From 5-PDB_NS.R
S_mutation_df = fread(file = "./Breast-S_freq-UP-PDB.csv", sep = ",")#From 6-PDB_S.R

library(dplyr)
# Mutant counts
mutcount = mutcount[order(-mut_count_NS,mut_count_S,length),]

# Gene - PDB_Chain data frame
gene_PDB_df = data.frame(gene = NS_mutation_df$Hugo_Symbol)
gene_PDB_df$PDB_chain = paste(toupper(NS_mutation_df$best_pdb),NS_mutation_df$Chain, sep =
"_")
gene_PDB_df = unique(gene_PDB_df)
gene_PDB_df = gene_PDB_df[!gene_PDB_df$PDB_chain == "NONE_none",] (#4317 rows,
# there are 4289 PDB_Chain files, some of them (28) are used for different genes: CKMT1A
and CKMT1B -> 1QK1_A)
index = match(mutcount$PDB,gene_PDB_df$PDB_chain)

######################################################################
# 3D cluster score computation
######################################################################
# Match mutant counts with genes
mutcount$gene = (gene_PDB_df$gene[index])

# CREATE new data frame for score calculations
file_size = nrow(mutcount)
score_file <- data.frame(index = 1:file_size)

score_file$gene = mutcount$gene
score_file$PDB = mutcount$PDB
score_file$cluster = mutcount$cluster
score_file$name = paste(mutcount$PDB,mutcount$cluster,sep = "_" )
score_file$mc_NS = mutcount$mut_count_NS
score_file$mc_S = mutcount$mut_count_S
score_file$length = mutcount$length
```

```
NSin = mutcount$mut_count_NS
Sin = mutcount$mut_count_S

cases_NS = unique(NS_mutation_df$Total_cases_GDC)
cases_S = unique(S_mutation_df$Total_cases_COSMIC)

# ------------------------------------------------
# S and NS frequency computation

NStotal = 2*cases_NS*mutcount$length
Stotal = 2*cases_S*mutcount$length
cs_NS_v = (NSin/NStotal)
cs_S_v = (Sin/Stotal)

# Save values in a data frame
score_file$cs_NS = cs_NS_v
score_file$cs_S = cs_S_v
# ------------------------------------------------
#Add 1 pseudo-count
NSin1 = NSin  + 1
Sin1 = Sin + 1
cases_NS1 = cases_NS + 1
cases_S1 = cases_S + 1

NStotal1 = 2*cases_NS1*mutcount$length
Stotal1 = 2*cases_S1*mutcount$length

# S and NS frequency computation and save results in vectors
cs_NS1_v = (NSin1/NStotal1)
cs_S1_v = (Sin1/Stotal1)
ratio1_v = (cs_NS1_v/cs_S1_v)
logratio1_v =log2(cs_NS1_v/cs_S1_v)

# Save values in a data frame
score_file$cs_NS_1 = cs_NS1_v
score_file$cs_S_1 = cs_S1_v
score_file$ratio_1 = ratio1_v
score_file$log2_ratio_1 = logratio1_v

# ------------------------------------------------
#Add 0.5 pseudo-count
NSin5 = NSin  +0.5
Sin5 = Sin +0.5
cases_NS5 = cases_NS + 0.5
cases_S5 = cases_S + 0.5

NStotal5 = 2*cases_NS5*mutcount$length
Stotal5 = 2*cases_S5*mutcount$length

# S and NS frequency computation and save results in vectors
cs_NS5_v = (NSin5/NStotal5)
cs_S5_v = (Sin5/Stotal5)
ratio5_v = (cs_NS5_v/cs_S5_v)
logratio5_v =log2(cs_NS5_v/cs_S5_v)

# Save values in a data frame
score_file$cs_NS_5 = cs_NS5_v
score_file$cs_S_5 = cs_S5_v
score_file$ratio_5 = ratio5_v
score_file$log2_ratio_5 = logratio5_v
```

```
# ------------------------------------------------
#Add 1e-10 pseudo-count
NSine = NSin  +1e-10
Sine = Sin +1e-10
cases_NSe = cases_NS + 1e-10
cases_Se = cases_S + 1e-10

NStotale = 2*cases_NSe*mutcount$length
Stotale = 2*cases_Se*mutcount$length

# S and NS frequency computation and save results in vectors
cs_NSe_v = (NSine/NStotale)
cs_Se_v = (Sine/Stotale)
ratioe_v = (cs_NSe_v/cs_Se_v)
logratioe_v = log2(cs_NSe_v/cs_Se_v)

# Save values in a data frame
score_file$cs_NS_e = cs_NSe_v
score_file$cs_S_e = cs_Se_v
score_file$ratio_e = ratioe_v
score_file$log2_ratio_e = logratioe_v

######################################################################
# Save Data
######################################################################

write.csv(score_file, "./Score.txt", row.names=FALSE) # RESULTS
print(paste('Total number of clusters:',nrow(score_file)))

end.time <- Sys.time()
time.taken <- end.time - start.time
print('Done!')
print(paste('Time:', time.taken, 'seconds'))
```